:UMENTATION PAGE

Form Approved
OMB No. 0704-0188

AD-A264 833

2. REPORT DATE: 11-24-9[illegible] FINAL [illegible] 1 NOV 90 - 31 [illegible] 9[illegible]

4. TITLE AND SUBTITLE: "COMPUTATIONAL FLUID DYNAMICS RESEARCH ON DYNAMICALLY ADAPTIVE MESH METHODS FOR TRANSONIC FLOWS"

5. FUNDING NUMBERS: 61106F; 2307/A1; AFOSR-90-0034

6. AUTHOR(S): RICHARD G. HINDMAN

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES): IOWA STATE UNIVERSITY, AMES, IOWA 50011

8. PERFORMING ORGANIZATION REPORT NUMBER: AFOSR-TR- 93 0205

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES): AFOSR/NA, BOLLING AFB DC 20332-6448

10. SPONSORING / MONITORING AGENCY REPORT NUMBER: AFOSR-90-0034

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT: Approved for public release, distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Dynamic mesh adaptation strategies are investigated. These include software development, dynamically adaptive mesh schemes, errors arising from generalized mappings and orthogonality. The strategies developed are tested against the euler and viscous Burgers Equations.

93 5 1[illegible] 031

93-10734

14. SUBJECT TERMS: CFD, ADAPTATION, MESHES

15. NUMBER OF PAGES: 48

16. PRICE CODE

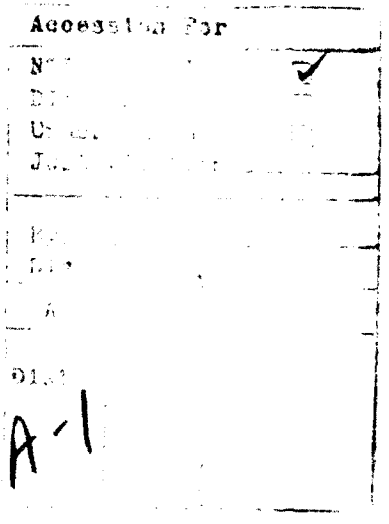
17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	[illegible]

FINAL REPORT
AFOSR-90-0034
MIPR ARO 149-89
DATE: 12/31/91

Prepared by: Richard G. Hindman
Associate Professor of Aerospace Engineering and Engineering Mechanics
2102 Black Engineering Building
Iowa State University
Ames, Iowa 50011
ph:(515) 294-0096
fx:(515) 294-8584

ABSTRACT

The effort reported on herein has concentrated on various aspects of computing unsteady flows on structured dynamically adaptive meshes. Various ways of coupling the mesh motion to the flow solution were studied. Many studies were performed on the one-dimensional unsteady Euler equations. Without the complexity of multidimensional phenomena (both physical and numerical), some clear trends and characteristics were identified relating to accuracy of the computed solutions and coupling methodologies. The results were somewhat surprising. Unsteady flows may be driven, in a stationary domain, by varying a boundary condition in time. They may also be driven by varying one or more physical boundary positions in time, thus simulating a moving wall, for example. This work has looked at both types of problems. The moving wall problems examined were both one and two-dimensional cases. Issues examined include the maximum allowable time step as well as solution accuracy as a function of amplitude and frequency of boundary point motion. The grid point positions in all cases were obtained as a solution to the Euler-Lagrange equations resulting from a variational formulation. The grid point speeds were obtained either by backward differencing the grid point positions or solving the grid speed equations obtained by differentiating the grid equations temporally. Two methods of solving these grid speed equations with differing levels of difficulty were employed in the one-dimensional problems considered. Two-dimensional problems studied include the unsteady flow about an oscillating wedge body, and unsteady flow through a CD nozzle. Some preliminary experience in computing flows with a shock impinging on a thin boundary layer region is also presented.



CONTENTS

2. Corresponding Mesh Speed Law

E. Appendix Figures

I. INTRODUCTION

This AFOSR/ARO funded effort here at Iowa State University started on November 1, 1989 and ended on October 31, 1992. The project was jointly funded by AFOSR and ARO and administered by AFOSR. Early discussions with both AFOSR and ARO scientists resulted in a mutual concurrence to direct the research effort toward the eventuality of the simulation of a moving flexible flight vehicle/projectile. The unsteady nature of both the fluid flow and the geometry in this case necessitates the use of, and a thorough understanding of, a dynamically adapting mesh. The mesh for such a problem must adapt in a variety of ways. It must adapt to accommodate boundary surface motion as well as to provide local refinement for optimizing solution accuracy. Boundary surface motion may be a result of rigid body movement of all or part of the surface and/or it may be a result of local deformation of surface elements. The significance of a completely time accurate algorithm for this type of simulation cannot be overstated. Issues relating to the feasibility and accuracy of this type of simulation are the subject of this effort and of this report.

This final report begins with a review of the project goals. Following this review, a summary of effort completed is presented which is primarily chronological. Conclusions are presented which represent a summary of all significant findings as they are expected to relate to the stated direction of this research. Details of various aspects of the total effort are presented in the appendix.

II . REVIEW OF CONTRACT GOALS

This section presents a list summary of the goals defined in the original proposal and presented by year for the reader's convenience.

A . Year 1

1) Develop animation software to graphically view unsteady motion of solution and grid
2) Develop dynamically adaptive mesh scheme for the wave equation
3) Develop mesh scheme for the inviscid and viscous Burgers' equation
4) Examine errors due to generalized mappings
5) Examine errors associated with mesh orthogonality
6) Examine ability to track shocks
7) Begin development of unsteady Euler and Navier-Stokes codes

B . Year 2

1) Develop mesh scheme for one-dimensional Euler equations
2) Examine errors in adapting to shock tube / blast wave problems
3) Develop mesh scheme for two-dimensional Euler equations
4) Examine ability to dynamically adapt to two-dimensional shocks

C . Year 3

1) Develop mesh scheme for two-dimensional Navier-Stokes equations
2) Implement turbulence model
3) Solve flat plate flow and adapt to boundary layer gradients
4) Solve shock-boundary layer impingement problem
5) Solve unsteady, transonic inlet problem

III . CHRONOLOGICAL SUMMARY OF EFFORT

A . Graphics Software

The first task was to develop appropriate simulation software for visualization of unsteady results. This was actually an ongoing task which grew in complexity as the need to visualize more data grew. At the present time, the simulation software runs on an Apollo workstation network and provides graphical output of grid and solution evolution for the 1-D and 2-D Euler and Navier-Stokes equations. The graphical output is in the form of a computer animation of a selected sequence of frames or a frame-at-a-time view of a user chosen variable.

B . Adaptive Mesh On 1-D Scalar Problems

A grid adaption philosophy was presented in the proposal. Testing of this philosophy required test cases for which an exact solution was known. This required that the dynamically adaptive mesh algorithm be implemented on some simple 1-D scalar model problems. The focus here was in the adaption of the mesh to solution changes as opposed to adaption to boundary point motion. The adaption procedure chosen was one based on a variational formulation which was an extension of the static scheme of Brackbill and Saltzman[1] into the time domain by deriving and solving a companion grid-speed equation simultaneously with the unsteady model equation. A similar approach based on a set of Poisson equations was presented in [2 - 4].

The need to begin the study at the scalar 1-D level was motivated by the reality that in order to guarantee success in the multi-dimensional problems of the real world, it is essential that a complete understanding of the effect of introducing so many additional degrees of freedom (doubled for the scalar 1-D problems and tripled for scalar 2-D problems) into the numerical problem is achieved. The effect on numerical stability, step size limitations, accuracy, wave formation, wave propagation, etc. were of interest. Previous work has mainly adopted a "let's see if we can do it" attitude without an exhaustive accuracy assessment based on known exact solutions and/or experimental data. We now know we can do it. What we do not know is how good our answer is, and what things affect this answer, and in what ways. In fact, in general, we do not even know how to define "good" in the present context for problems whose exact solutions are not known. Consequently, the present effort sought to quantify these issues to the extent possible.

1 . Numerical Scheme

A number of test cases were coded and run for the wave equation and Burgers' equation which included moving linear and non-linear fronts as well as periodic waves. The integration algorithm used was the standard MacCormack explicit scheme. This was chosen as a baseline algorithm due to its popularity, ease of extending to 3-D, and ease and computational efficiency of implementing as a TVD scheme [5 - 6] (Note: the upwind scheme of Warm-

ing and Beam [7] was also used for one test). The basic algorithm for the coupled solver is given as follows:

1) Given u(x) at time 0, and an initial mesh x_j

2) Apply the grid law at time 0 to generate a mesh x_j consistent with u(x)

3) Apply the grid-speed equation at time 0 to generate $(x_\tau)_j$

4) Apply the predictor step of the algorithm to predict u_j and x_j at the next time, $\tau + \Delta\tau$

5) Apply the grid law at this time to generate a mesh x_j consistent with this u_j

6) Apply the grid-speed equation at this time to generate $(x_\tau)_j$

7) Apply the corrector step of the algorithm to improve u_j and x_j at this time, $\tau + \Delta\tau$

8) Apply the grid law at this time to generate a mesh x_j consistent with this corrected u_j

9) Apply the grid-speed equation at this time to generate $(x_\tau)_j$

10) Go to step 4)

A modified algorithm was also tested and used for many of the computations. In the modified algorithm, steps 5 and 8 are eliminated and steps 6 and 9 are replaced by the line:

6) Apply the augmented grid-speed equation at this time to generate $(x_\tau)_j$

where the word "augmented" is used to indicate the use of the grid speed control law approach. More details of this approach are presented in the 2-D context in subsection F 3 of this section. The essence of it is that the grid law and the grid speed law are combined into one equation with feedback control characteristics such that grids obtained by integrating grid speeds maintain there form as solutions of the grid law without having to solve the grid law explicitly. This saves significant CPU time.

Equations of the form: $\frac{d}{dt}\int_R u\,dx + \int_{\partial R} df + \int_R h(x)dx = 0$ are solved in a finite-volume context on a generalized mesh with moving control volumes using a TVD version of MacCormack's scheme. Here, $f = f(u)$ and $h(x)$ is a prescribed forcing function which is sometimes used to make the problems yield solutions which tend to non constant functions in the steady state. The grid law for these 1-D problems has the form:

$$x_{\xi\xi} + x_\xi\left(\frac{w_\xi}{w}\right) = 0\,. \qquad (1)$$

As suggested in [8 - 9], the dependent variable in this equation may be the arc length along the function w(x), which is called the *monitor surface*. The grid speed equation corresponding to this grid law which must govern the proper mesh dynamics is given by:

$$(x_\tau)_{\xi\xi} + \left(\frac{w_\xi}{w}\right)(x_\tau)_\xi + \left(\frac{w(w_\tau)_\xi - w_\xi w_\tau}{w^2}\right)x_\xi = 0 \qquad (2)$$

The numerical solutions to these coupled equations for the test problems indicated were compared directly to the exact solutions. The exact error provided precise information as to the acceptability of the various weight functions and corresponding user input parameters.

2. Coupling Methods

The finite-volume discrete approximation to the PDE equation has grid speed in it as part of the flux evaluation. The grid speed equation has grid speed as well as gradients of some user selected function which eventually depends on gradients of the PDE solution. These two equations are inherently coupled. Three basic methods of solving these equations were examined in this work. They are called: 1) differenced speed (x_τ), 2) loosely coupled (differenced w_τ), and 3) strongly coupled. These methods represent drastically different degrees of difficulty. Method 1 is by far the simplest and, in fact, removes the need of even having a grid speed law. The grid speeds are simply determined by a simple backward difference of the grid point positions. Method 2 solves the grid speed law but the quantity, w_τ, appearing in the equation is computed from a backward difference. Method 3 is extremely complex in that the w_τ is replaced by the implied sum $w_\tau = w_{u_i} u_{i\tau}$ where $u_{i\tau}$ is replaced by the difference approximation to the governing PDE at the point i. This renders the grid speed equation completely linear (except for the TVD terms) in the grid speeds and allows a solution for the grid speeds with the PDE approximation and all of the associated physics of the subject problem to influence the results in a time accurate fashion. See the appendix for a more thorough discussion of the coupling procedures.

3. Strong Coupling

The strong coupling procedure results in the solution of a nine-diagonal system of equations. Substantial effort was required in developing this system and finding an efficient way of solving it. The system character is a result of using a smoothed solution to drive the mesh. This is necessary for problems which contain abrupt solution features such as shocks. Two smoothing passes result in a nine-diagonal system whereas three passes produce an eleven-diagonal system. Two passes were sufficient for the problems studied.

4. Weight Function

The mesh adaption scheme used in this work requires a minimum of two user input parameters which control the amount of adaption for a given weight function. The weight function is, however, also a user choice. For example, the user may elect to adapt to regions where the magnitude of the gradient of a chosen function is large. Or, it may be more desirable to adapt to regions with large second derivative, or perhaps curvature, or a combination of these things. Such a generalized weight function is expressed mathematically as:

$$w = 1 + a\,g_1 + b\,g_2 + c\,g_3 + d\,g_4 + \text{etc.} \qquad (3)$$

where the coefficients, (a,b,c,d,...) are user prescribed and the g_i's are squares of gradients, second derivatives, curvatures, etc. Even after this decision of what to adapt to is made, the user must input parameters indicating how much to adapt. While some work has been done on trying to automate the selection of the input parameters for certain types of weight functions [10], little work (with notable exceptions [8 - 9]) has been done on trying to determine the best form for the weight function. Because of this, the present effort was initially concentrated on these issues.

5. Some Accuracy Tests

The error in the numerical solutions at a point is defined as the difference between the computed result at the point and the exact result at the point. Another measure of error is obtained by integrating these pointwise errors across the mesh and dividing by the mesh length. This is, of course, a more global measure of error. In problems with shocks, this global error can be totally dominated by either shock speed error or overshoot error when TVD schemes are not used. Because of this, care in the interpretation of this error measure must be exercised.

Tests for these 1-D model equations were conducted as indicated and the results were initially puzzling. Results indicated the following general conclusions.

1) The adaptive grid solution was not always better than the equivalent fixed grid solution.

2) The adaptive grid solution was sometimes unstable when the fixed grid solution was not.

3) The adaptive grid solution could always be made better than the fixed grid solution if enough effort was spent choosing the weight function and the adaption intensity parameters but the exact solution was required to guide these user choices.

4) The user choices for weight function and adaption intensity parameters which provide the best result for one problem do not, in general, provide the best result for a different problem.

5) The user choices for weight function and adaption intensity parameter which provide the best result for one numerical algorithm applied to a given problem do not necessarily provide the best result for a different algorithm applied to the same problem.

The results of the indicated accuracy tests proved that simply being able to move the mesh points dynamically into regions with prescribed features as the solution changed did not necessarily produce a numerical solution which was better than that computed on a fixed mesh. This meant that there was something missing in the understanding of what was actually needed to constitute a "best mesh" or a "good grid" as it is sometimes referred to. Consequently, the focus of the effort turned to this issue.

C. Errors Due To Generalized Mappings

The following path of reasoning was set forward to address this issue: Suppose a problem of interest requires the solution of a partial differential equation (PDE) which contains a derivative term, f_x, where $f = f(u(x))$ with $u(x)$ representing the dependent variable of the PDE. The numerical solution of such a problem requires first a discretization of the x-domain into a sequence, say, $\{x_i, i = 0,1,...,I\}$ subject to the constraint that $x_0 < x_1 < x_2 < ... < x_I$, and second a suitable numerical approximation to the derivative, f_x. The objective of the choice of the particular sequence $\{x_i\}$ and the particular discretization used in approximating f_x is clearly to minimize the error between the numerical value of f_x and the exact value of f_x at each of the mesh points. Clearly then, a mesh which performs this function is a "good mesh". Moreover, a mesh/discretization combination which produces zero error is the "best mesh" possible. A numerical procedure was then developed to obtain this "best mesh" for specified functions and derivative approximations. In the original formulation, the actual function, $f_x(x)$, was required for the numerical procedure suggested. Details of this are found in the appendix. Later work, however, revealed that a spline fit of data representing the function was adequate for near zero error. At any rate, the value of the procedure lies not specifically in its practical applicability (the extent of which remains to be demonstrated), but rather in its ability to produce categorically the "best mesh" for any differentiable function for which a first derivative expression is known. This does, at a very minimum, provide information about what a "good grid" should look like and what characteristics it should have. This information, however, is valid only for the particular discrete approximation used, which is, in most cases studied, a central difference. It is possible to use other approximations also such as a second order backward difference The "best mesh" in this case is, strictly speaking, different than the "best mesh" in the central difference case. The important thing to recognize here is that the "best mesh" depends upon both the *nature of the function* f(x) and the *choice of discretization procedure* of the derivative, f_x. Adaptive mesh procedures tc date have focused on only the first of these. In theory, one could determine the "best mesh" to resolve numerical approximations to other types of terms also, such as the quantity: $f_x - \mu u_{xx}$ found in the viscous Burgers' equation. In this case, however, other conditions besides no inflection points must likely be imposed to guarantee uniqueness of the result, if such uniqueness is even possible.

The practical usefulness of the idea suggested here is unknown for multi-dimensional multivariable problems. Preliminary work shows that "best grids" may be obtained in 2-D for some problems. In any case, the lessons learned to date are valuable guides in helping to formulate new weight functions and new adaption schemes which incorporate information regarding both the *nature of the function* f(x) and the *discrete approximation used* to represent its derivatives.

D . Extension To 1-D Unsteady Euler Problems

The basic grid and grid-speed laws derived for the scalar model problems are the same as those for the 1-D Euler equations. Various methods of coupling the mesh dynamics equations and the fluid dynamics equations are possible as mentioned. These methods were re-

ferred to as methods 1, 2, and 3. Experiments were conducted with various differencing strategies for these methods. The work was first focused on 1-D shock tube computations for the problem described below. In these computations, the weight function chosen was the square of the density gradient plus one. This caused adaption to both the moving shock and the contact discontinuity.

1. Initial Conditions

The initial conditions for the shock tube problem were:

$$U_L = \begin{Bmatrix} \varrho \\ \varrho u \\ \varrho e_t \end{Bmatrix} = \begin{Bmatrix} 0.445 \\ 0.311 \\ 8.928 \end{Bmatrix} \qquad U_R = \begin{Bmatrix} \varrho \\ \varrho u \\ \varrho e_t \end{Bmatrix} = \begin{Bmatrix} 0.5 \\ 0.0 \\ 1.4275 \end{Bmatrix}$$

These were chosen to provide for a comparison with the results in [6] should the need arise during the debugging stages of code development.

2. Adaption Errors For Shock Tube

In order to continually assess the quality of an adaptive mesh solution, certain performance criteria were established. The absolute pointwise error is the difference between the computed solution at a mesh point and the exact solution at that point. The global error is just the integrated pointwise error across the domain. In problems with shocks, this global error can be totally dominated by either shock speed error or overshoot error when TVD schemes are not used or are used improperly. Because of this, care in the interpretation of this error measure must be exercised.

3. Results for Stationary Boundaries

The findings are conclusive in some respects but misleading in others, for the cases studied which involved only solution adaption as opposed to boundary adaption. For problems of this type, the strongly coupled method (method 3) clearly requires the most computational effort. The differenced speed method (method 1) clearly produces the most accurate results of the computations performed with an adapting grid. Figure (1) shows a comparison of shock tube solutions for this case, versus the exact solution, using a non-adapting uniformly spaced grid, and adapting grids using methods 1, 2, and 3. In this figure, it is clear that method 1 produces a solution of comparable accuracy (in appearance) to the solution obtained on the uniform unadapted mesh. It also appears that method 1 produces better results for this case than either of methods 2 or 3. It was of more than casual interest that the solution on a uniform non-adapting mesh appeared to be as good as the best adaptive grid solution. Indications on the shock tube problem were that if enough points were used, a uniform mesh solution was more accurate than a strongly coupled adaptive mesh solution. It appeared that there was some

critical number of points below which the strongly coupled adaptive mesh solution was better than the uniform mesh solution. For the shock tube problem studied, this critical number of points was around 80.

Figures (2) and (3) support this notion. Figure (2) illustrates the integrated error in density versus time for the same shock tube calculation but with 101 points and with all three mesh speed strategies. The one labeled "backward P_tau" refers to the loosely coupled scheme of method 2, while the one labeled "backward X_tau" refers to the differenced speed method (method 1). It is apparent from this figure that the fully coupled method produces the most error. Interestingly, this method also takes the most CPU time. The smallest error and smallest CPU time occurs for the differenced speed method. Note that this result is for an adaption constant of 3. In these figures, the error resulting from no adaption is presented for a reference. The minimum error in both of these figures occurs for the differenced speed method. As pointed out, the most advantage from an adaptive mesh scheme should come from computations on a coarse mesh. Figure (3) illustrates the same computation as Fig. (2) except 51 points are used instead of 101. It is clear from these figures that improved solution accuracy relative to the uniform mesh accuracy is possible with adaption on a coarse mesh. However, the benefit may not be worth the price in CPU time. It should be noted that a typical adaptive mesh solution took 0.0626 sec/step as compared to 0.0143 sec/step for the uniform mesh case. In addition, due to the step size restriction imposed by the finer mesh, it took 2.4 times as many steps to reach the same time. The results here make it clear that there is no real payoff for this problem in using an adaptive mesh based on a weight function of density gradient. Better weight functions undoubtedly exist and substantially more improvement is possible for other types of problems. However, it is very encouraging that the adaptive mesh scheme does an excellent job of accurately tracking shock position and strength.

It would seem that the differenced speed method is the method of choice based on these comparisons. As mentioned earlier, these results are valid only for the case of solution adaption to flows with shocks. If the boundaries of the domain of interest were set into motion, these conclusions need not apply. This possibility was tested.

4. Results for Moving Boundaries

The results in the previous section were bothersome. Clearly, mesh adaption with all of its overhead did not improve the results enough to justify its use. The previous results dealt with solution adaption. The boundaries were not moving. A decision was made to study the effect of setting the boundaries in motion. Two philosophies for doing this were apparent. An oscillation of the boundaries could be interpreted as a physical disturbance to the geometry or it could be viewed as simply a movement of the computational domain boundaries on the non-moving geometry. This latter scenario was chosen for the test case. Each boundary of the shock tube was given a prescribed sinusoidal position description. For the left end, for example, the left boundary point moved in time according to:

$$x_1(\tau) = M \sin(\omega\tau + \phi) \qquad (4)$$

where M, ω, and ϕ were user inputs. The right boundary was treated similarly. Typically, M was set to some constant times the equivalent uniform Δx. The input value of ω was chosen in accordance with T_{max} by requiring a selected number of cycles per simulation time. The value of ϕ was 0 for all cases considered. Many computations were performed with method 1 and method 2 (several hundred cases). From the data generated, the integrated error in density at the final time was used as the measure of accuracy of the computed solutions. This integrated error is plotted versus frequency with amplitude as a parameter in Fig. (4), and versus amplitude with frequency as a parameter in Fig. (5). The amplitude is M in these figures and the frequency is the number of cycles in the fixed amount of time covered by the computations. The major conclusion from these figures is just as suspected. The benefit of the greater coupling between the grid speed equations and the flow equations increases as the boundary motion increases. The error present at the lower frequencies and magnitudes is simply a result of the truncation error of the numerics used to approximate the grid speed equations. This is in contrast to the error in obtaining the grid speeds by simple backward difference. As the integration proceeds, the simple backward difference of method 1 allows a build-up of error which could be interpreted as round-off. The higher coupling of method 2 inhibits this build-up and thus, at some frequency and magnitude, method 2 becomes a more accurate approximation.

This completes the considerations for 1-D computations. The next topic is the extension to 2-D.

E . 2-D Considerations

Other considerations, not discussed with respect to the 1-D examples presented, are important in extending to multi-dimensional problems. One of the most obvious is the issue of orthogonality.

1 . Errors Associated With Mesh Non-Orthogonality

The basic concern was the error introduced into a computation as a result of non-orthogonality of the mesh. In a dynamically adapting mesh, this concern was clearly greater since the real possibility exists that cell collapse may be inadvertently driven by the adaption scheme. An attempt was made to study the way in which non-orthogonality influences the numerical approximation.

a . Procedure and results

The basic procedure, outlined here and detailed in the appendix, was to represent the derivative f_x in a 2-D generalized coordinate frame with derivatives in the generalized coordinate directions approximated by central differences. The f quantities were then expanded about

the reference point resulting in an equation of the form:

$$f_x = C_1 f_x + C_2 f_y + C_3 f_{xx} + C_4 f_{yy} + C_5 f_{xy} + \ldots$$

An analysis was performed to see how the grid structure locally and the metric evaluations influenced the C_i's. In summary, the basic results were that:

1. metrics should be evaluated the same as f_ξ , f_η to zero C_2 and make C_1 unity.
2. minimizing coordinate line curvature and stretching also minimizes numerators of C_3, C_4, C_5.
3. minimizing non-orthogonality maximizes denominators of C_3, C_4, C_5.

These basic results are expected to be applicable to upwind difference approximations as well since they appear to be common sense in nature.

F . Dynamic Adaption For 2-D Euler Equations

The basic grid and grid-speed laws for 2-D problems were derived. More details are found in the appendix. It must be noted that the equations introduced by [1] are dimensionally inconsistent unless dimensions are absorbed into the user input parameters. This was found to be inappropriate resulting in difficulties setting the values of the parameters for a desired level of grid adaption. The parameter values for a given level of adaption were problem dependent. The effectiveness of various terms in the performance index was found to vary over the domain in problems with a significant clustering requirement. This caused the effect of a given term, say orthogonality, to be more significant in some regions than it was in others. The cure for this scaling problem was to redefine the performance index in a dimensionally consistent manner with user input parameters of order 1. The Euler-Lagrange equations were then re-derived for this case and the grid speed equations were derived by differentiating these new Euler-Lagrange equations with respect to the temporal variable.

1 . Modified Mesh Law

The result of the derivation was a second order PDE system of the form:

$$G(\vec{r}) = A\vec{r}_{\xi\xi} + B\vec{r}_{\xi\eta} + C\vec{r}_{\eta\eta} + D\vec{r}_{\xi} + E\vec{r}_{\eta} = 0 \qquad (5)$$

which constitutes the grid law. The boundary conditions are mixed Dirichlet and Neumann depending upon the physical conditions required of the mesh at the boundaries.

This new mesh law requires specification of 3 parameters, λ_s, λ_o, and λ_a corresponding to smoothness, orthogonality, and adaption terms. Note that in the new formulation, these may be functions of time. This may be useful for problems in which the flow structure being adapted to diminishes with time while the user wishes to maintain resolution at this location during the decay process. These parameters are nondimensional and have the value of unity if orthogonality and solution adaption are equal in importance to smoothness. In the old scheme, the values of these parameters were functions of the problem, the number of mesh

points, the distribution of points, etc., which required experimentation with various values in order to discover an acceptable set (if one even existed).

2. Corresponding Mesh Speed Law

The result of the temporal differentiation was a linear, variable coefficient, second order PDE system of the form:

$$G_\tau(\vec{r}) = A\vec{z}_{\xi\xi} + B\vec{z}_{\xi\eta} + C\vec{z}_{\eta\eta} + D^*\vec{z}_\xi + E^*\vec{z}_\eta + \vec{T}^* = 0 \qquad (6)$$

which constitutes the grid speed law. In this equation, $\vec{z} = \vec{r}_\tau$, the grid speed vector. The boundary conditions are mixed Dirichlet and Neumann for this equation as well, depending upon the physical conditions required of the mesh speed at the boundaries.

3. Augmented Mesh Speed Law

The mesh law and mesh speed law were combined into one equation called the "augmented" mesh speed law. This equation is written symbolically as:

$$G_\tau(\vec{r}) + \lambda_C G(\vec{r}) = 0 \qquad (7)$$

where λ_C is a user specified feedback constant which was typically set to a number between 50 and 500. This number is of order $1/\Delta\tau$. This equation is used as a feedback control law which removes the need for solving the grid law. When the grid speeds obtained from solving this "augmented" grid speed equation are simply integrated in time to obtain the grid at the new time, this new grid automatically satisfies the grid law.

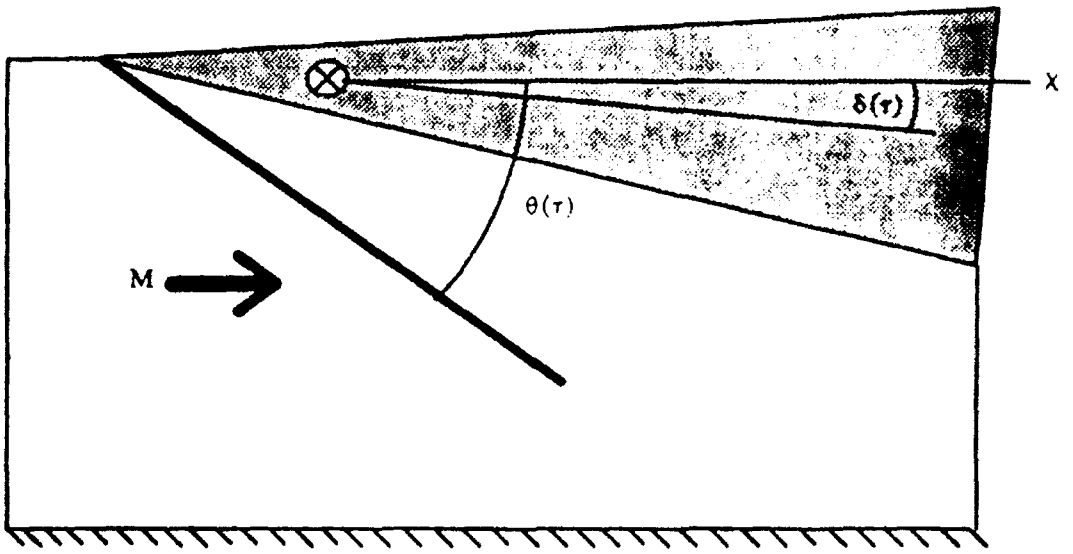
4. Solution Strategies

Based on the findings in 1-D regarding the strongly coupled procedure and based on the enormous complexity and CPU requirement of this method, it was rejected as a candidate for the 2-D computations. The two methods called methods 1 and 2 were employed. The TVD version of MacCormack's method was extended to 2-D and used as the numerical integrator.

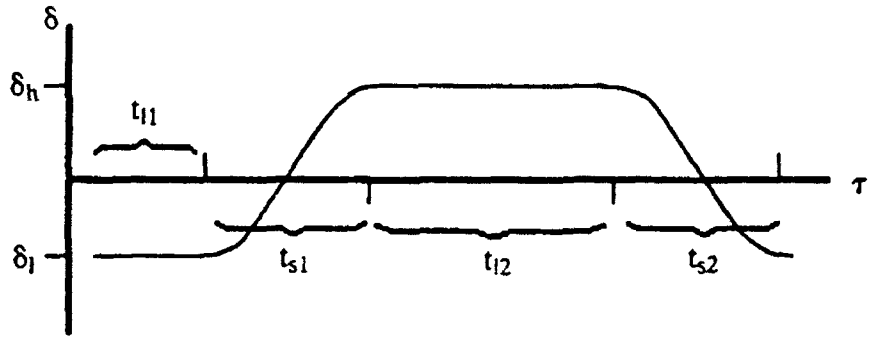
5. Oscillating Wedge

The first 2-D test on Euler equations is an oscillating wedge/plate interaction problem as shown.

The $\delta(\tau)$ is a user prescribed function as is the point about which rotation is to take place. The resulting shock pattern will normally oscillate between a regular reflection pattern and a Mach reflection pattern. The incident shock may be planar, curved, or non-existent (it could be an expansion) depending upon the magnitude of the oscillation. The triple point solution for the Mach reflection problem is known for steady problems. The regular reflection solution for the planar shock is also known for the steady problem. This information helped to validate some of the unsteady solutions.



The test run involved a general $\delta(\tau)$ as shown below.



The high and low values of δ were input as well as the duration times of each phase of the curve. This defines a general periodic input which can be a sinusoid as well as a more general function. An example of a simple compression-expansion process follows.

Results were computed but are not shown here for a 20 degree wedge in a Mach 2 flow with two schedules back to back which cause the wedge to start at $\delta_l = -10$ degrees which corresponds to no flow deflection on the lower side. The wedge then moves along a sine curve to $\delta_h = 0$ thus producing a maximum compression deflection of 10 degrees. This is followed by a retreat back to $\delta = -10$ degrees and then on to an expansion. The results of a computation of this sort involve over a thousand frames of data, many of which are viewed on the workstation as (in some cases, after) the solutions are computed. The management of this much data is an enormous task. Graphical procedures have been developed to enable the management and processing of this data. The data can now be made into an animation video for use in presentations.

One of the cases which was run and is presented here started the lower wedge surface for the 20 degree wedge parallel to the freestream and then the body was plunged downward until a prescribed compression angle on the lower surface was achieved. This case is similar to the previous one except that when the compression angle was achieved, the body remained at that angle. The unsteady flowfield in this case started as a parallel supersonic freestream and began a compression process which, after much shock development and interaction, ultimately caused the flow to reverse, first at the downstream boundary, and then proceed to the left. This process continued until the entire flow was reversed. This is, in effect, a simulation of the choking process in a supersonic flow. The animation of this flow evolution was indeed fascinating. A VHS format video of this animation was made and a copy of it is available upon request. It is not in color as the hardware available at the time prevented this possibility. Two pages of selected frames illustrating the flow evolution are presented in Figs. (6) and (7). The first of these illustrates Mach contours while the second shows velocity vectors. Progress is from left to right.

In the computation of this case, it became apparent that the explicit CFL condition alone was not always sufficient for the determination of the stable time step. Some effort was spent studying this in more detail. A list of the relevant time scales (on the mesh scale) was compiled. The two conventional ones which come from an Eigenanalysis, are given by:

$$\Delta\tau_x = \frac{\Delta x}{|u - x_\tau| + a} \quad , \quad \Delta\tau_y = \frac{\Delta y}{|v - y_\tau| + a} \tag{8}$$

In these equations, (u,v) and (x_τ, y_τ) represent the components of the velocity and the speed of the mesh point in the x,y directions respectively, and 'a' represents the acoustic velocity. A third time scale which seems intuitive for this problem is given by

$$\Delta\tau_\omega = \frac{\Delta y}{|V_{body}|} = \frac{\Delta y}{R|\omega|}$$

where ω is the angular velocity of the body motion and R is the distance from the pivot point to the point in question. This time scale is distinctly different than $\Delta\tau_y$ above since $v-y_\tau$ vanishes at the body. This time scale results from the restriction that the body motion can not cause a mesh point to sweep across more than one existing mesh cell in a single time step. This restriction is consistent with the notion of time accurate unsteady flow simulation. It appears that it is the relationship of this time scale to the smallest of the other two which determines if there is any advantage to using implicit methods. If $\Delta\tau_\omega$ is significantly larger than the minimum of the other time scales, then implicit methods are likely to be of significant benefit. A more general statement of this idea is:

$$\Delta\tau_{\omega x} = \frac{\Delta x}{|x_\tau|} \quad , \quad \Delta\tau_{\omega y} = \frac{\Delta y}{|y_\tau|} \tag{9}$$

and the minimum of these stepsizes determines the significant timescale to be compared with the conventional ones from eigenanalysis.

6 . CD nozzle

A 2-D computation of flow through a converging-diverging nozzle was made and compared with a solution of the model problem of quasi 1-D flow through a variable area duct. The case of interest is one with a combination of inlet/exit conditions which cause a shock to form just downstream of the throat. The exit flow Mach number is subsonic in this case which requires specification of one flow condition there. The variable chosen was static pressure. This exit pressure was varied sinusoidally thus causing the shock position to oscillate in the nozzle. An interesting aspect of this problem was that if the back-pressure oscillation was strong enough and rapid enough, it appeared that the pressure wave generated by this boundary condition was nearly a shock which traveled upstream and interacted with the shock already present in the nozzle.

G . Dynamic Adaption For 2-D Navier-Stokes Equations

Extension of the dynamic adaption procedures to viscous flows was performed. Some simple test cases were computed. These included a flat plate boundary layer calculation in both subsonic and supersonic flow, and a shock-boundary layer interaction problem reported on by Hakkinen et al. [11]. Results for this latter case are presented here.

1 . Considerations for Viscous Flows

Dynamic adaption for viscous flows was and remains a real challenge. Grid points are required to cluster near the wall in boundary layer regions while at the same time clustering to features in the outer flow such as shocks. It was unclear what weight function would provide the best of both worlds. One based on Mach number gradient was used.

Further complications arose due to the inadequacy of simple turbulence models to correctly model flow physics in cases with turbulence. This ever-present reality was exacerbated by the additional degrees of freedom present in the system of equations due to the dynamically adapting grid, and by the difficulty in getting the grid to adapt enough in the boundary layer region without over-adapting at the shock, even in laminar boundary layer problems where wall clustering requirements are significantly less than those for turbulent boundary layers..

2 . Shock-Boundary Layer Interaction

The supersonic flow over a flat plate with an impinging shock was studied in Ref. [11]. The flow conditions for this test case were: $M_\infty = 2$, $Re_\infty = 2.96(10)^5$, $T_\infty = 250°K$, and $L = 1$ me-

ter. The computations for this case were made on both a clustered static mesh and a dynamically adaptive mesh. The static mesh and the converged dynamically adaptive mesh are shown in Fig. (8). It is clear that there is significantly less clustering in the dynamically adapting mesh in the boundary layer region. Figure (9) illustrates the pressure contours for both the static mesh and the dynamically adaptive mesh. The shock resolution is clearly improved by the adaptive mesh. However, surface pressure, for example, shown in Fig. (10), demonstrates that the adaptive mesh solution misses the separation point and the experimental data, in general, more than the static mesh solution. It is felt that this discrepancy is mainly due to lack of resolution of the boundary layer region by the adapting mesh.

IV. CONCLUSIONS

Dynamic adaption of a mesh is a result of either boundary motion or solution adaption or both. Solution adaption may be a result or boundary motion, boundary condition unsteadiness, or just inherent solution unsteadiness. The present research results suggest that unsteadiness resulting from boundary motion may require a greater level of coupling between grid motion prediction and flow solution prediction than unsteadiness resulting from simple moving waves. It also is very clear that dynamically adaptive meshes are very CPU intensive and should be used only for problems on which there use is needed. Information is presented in this report which guides the researcher regarding when adaptive meshes are useful and when they just produce excess baggage. Information is also presented which guides the choice of weight function in the context of the definition presented for constructing the "best mesh."

This work tested some ideas regarding the significance of various time scales present in a flow problem with dynamic motion. The results from this testing suggest a means of guessing when it is beneficial to use implicit methods instead of explicit ones.

The various methods of coupling the mesh dynamics and the flow dynamics is, of course, a crucial issue. The strong coupling procedure is the most elegant, and the most rigorous. It does not produce the best results in the cases discussed in this report, but as mentioned, is expected to be superior for problems in which there is complex and/or rapid mesh motion due to complex flow feature evolution or complex boundary motion. Unfortunately, it is unquestionably the most expensive. Perhaps the greatest disadvantage of the strong coupling procedure is that changing the nature of the weight function requires a significant analysis effort and a non-trivial code modification effort. The loosely coupled procedure (method 2) does not suffer from this difficulty and consequently is the method of choice for problems requiring dynamic adaption.

The mesh adaption procedures explored in this research are ideally suited for use in problems with moving as well as deforming boundaries.

The work has been carefully guided to produce numerical procedures which are all extendable to 3-D. This extension is straightforward.

V. ACKNOWLEDGEMENT

The author wishes to thank first, Drs. Len Sakell and Tom Doligalski, the technical monitors on this project, for the helpful thoughts provided and second, the agencies of AFOSR and ARO for providing the funding to make this work possible.

VI. REFERENCES

1. Brackbill, J., and Saltzman, J., "Adaptive Zoning for Singular Problems in Two Dimensions," Journal of Computational Physics, Vol. 46, June 1982, pp. 342-368.

2. Hindman, R.G., Kutler, P., and Anderson, D., "Two-Dimensional Unsteady Euler Equation Solver for Arbitrarily Shaped Flow Regions," AIAA Journal, Vol. 19, April 1981, pp. 424-431.

3. Hindman, R.G. and Spencer, J., "A New Approach to Truly Adaptive Grid Generation," AIAA Paper 83-0450, AIAA 21st Aerospace Sciences Meeting, Reno, Nevada, January 1983.

4. Holcomb, J.E., and Hindman, R.G., "Development of a Dynamically Adaptive Grid Method for Multidimensional Problems," AIAA Paper 84-1668, AIAA 17th Fluid Dynamics, Plasma Dynamics, and Lasers Conference, Snowmass, Colorado, June 1984.

5. Davis, Stephen F., "TVD Finite Difference Schemes and Artificial Viscosity," ICASE Report No. 84-20 and NASA CR 172373, June 1984.

6. Causon, D.M., "A Total Variation Diminishing Scheme for Computational Aerodynamics," Numerical Methods For Fluid Dynamics III, New York: Oxford University Press, 1988, pp. 449.

7. Warming, R.F., and Beam, Richard M., "Upwind Second Order Difference Schemes and Applications in Aerodynamic Flows," AIAA Journal, Vol. 14, No. 9, September 1976, pp. 1241-1249.

8. Eiseman, Peter R., "Alternating Direction Adaptive Grid Generation," AIAA Journal, Vol. 23, No. 4, April 1985.

9. Bockelie, Michael J., and Eiseman, Peter R., "A Time-Accurate Adaptive Grid Scheme and the Numerical Simulation of a Shock Vortex Interaction," submitted to Journal of Computational Physics, October 1989.

10. Dwyer, H.A., "Grid Adaption for Problems in Fluid Dynamics," AIAA Journal, Vol. 22, December 1984, pp. 1705-1712.

11. Hakkinen, R.J., I GreBer, L. Trilling, and S.S. Abarbanel. "The Interaction of an Oblique Shock Wave with a Laminar Boundary Layer." NASA Memo 2-18-59W, 1959.

VII . FIGURES

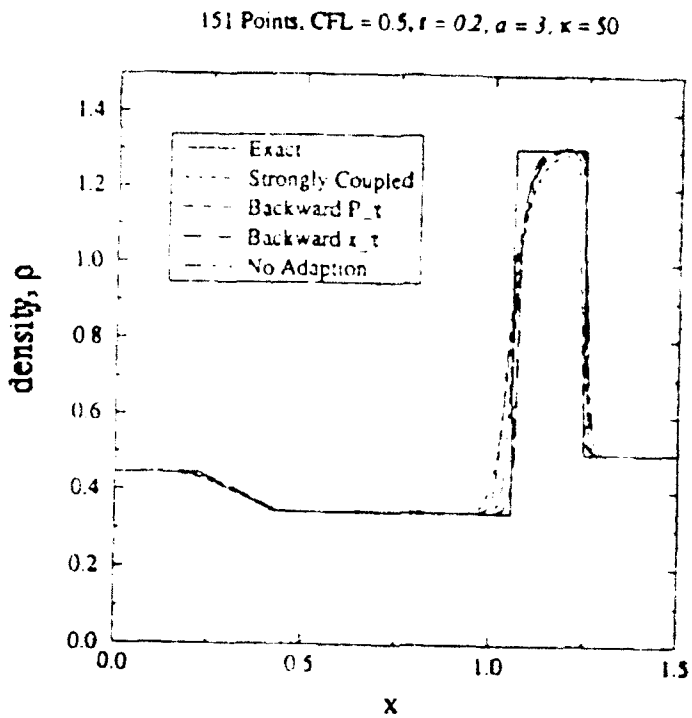


Figure (1). uniform, method 1, 2, and 3, and exact solutions.

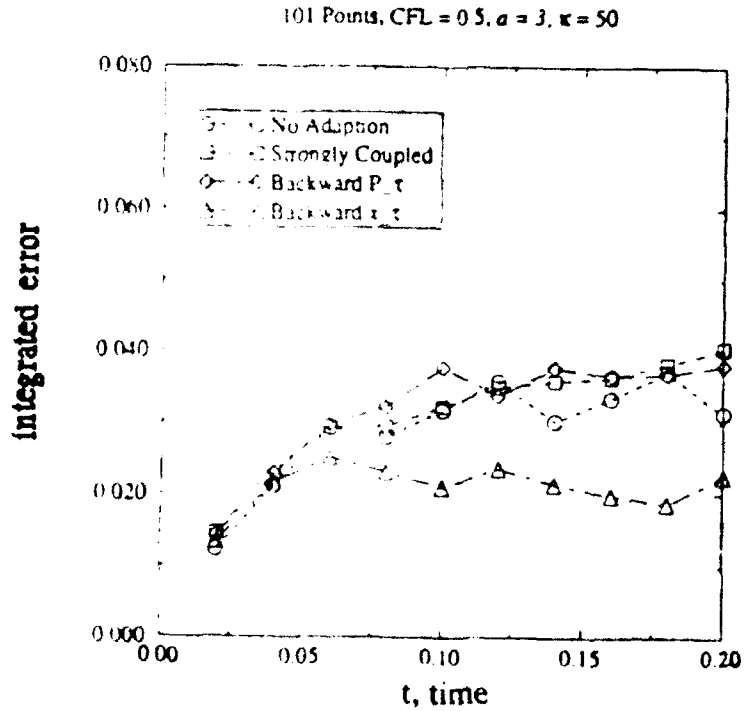


Figure (2). integrated density error versus time (101 points)

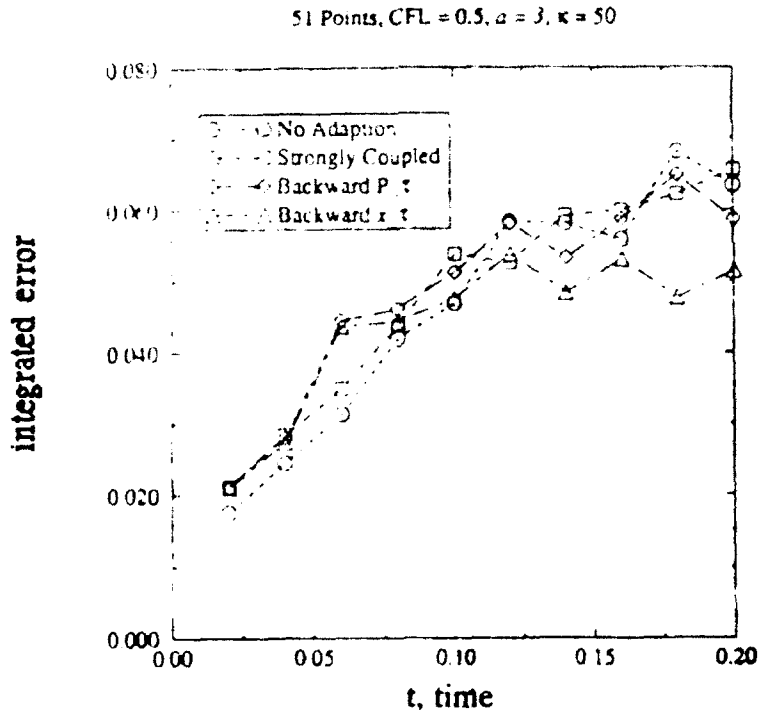


Figure (3). integrated density error versus time (51 points)

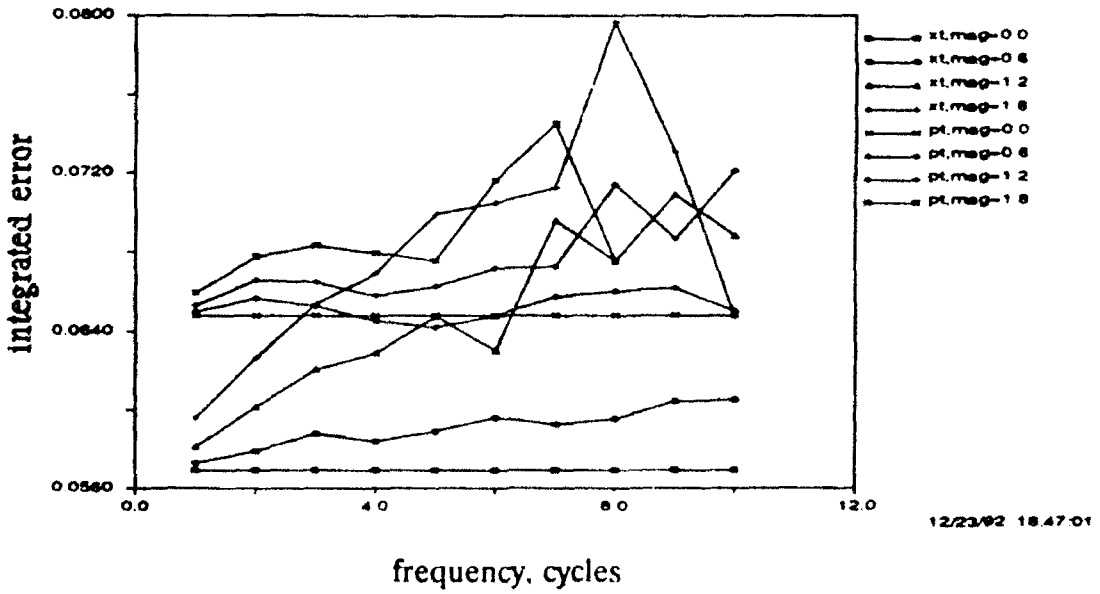


Figure (4). integrated density error versus oscillation frequency for methods 1 and 2

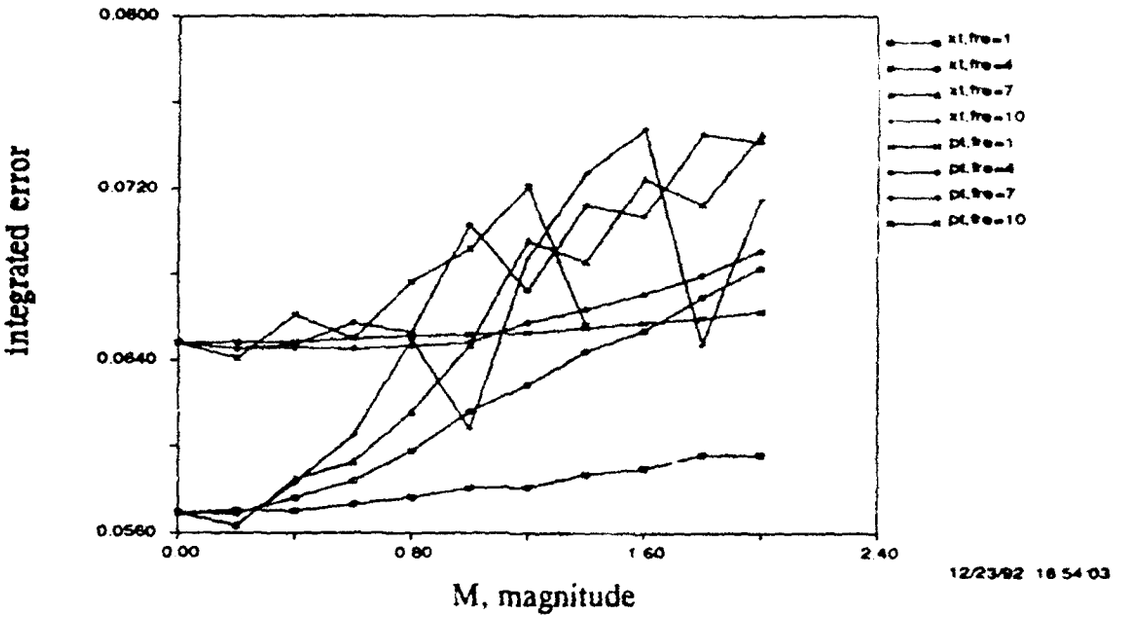


Figure (5). integrated density error versus oscillation magnitude for methods 1 and 2

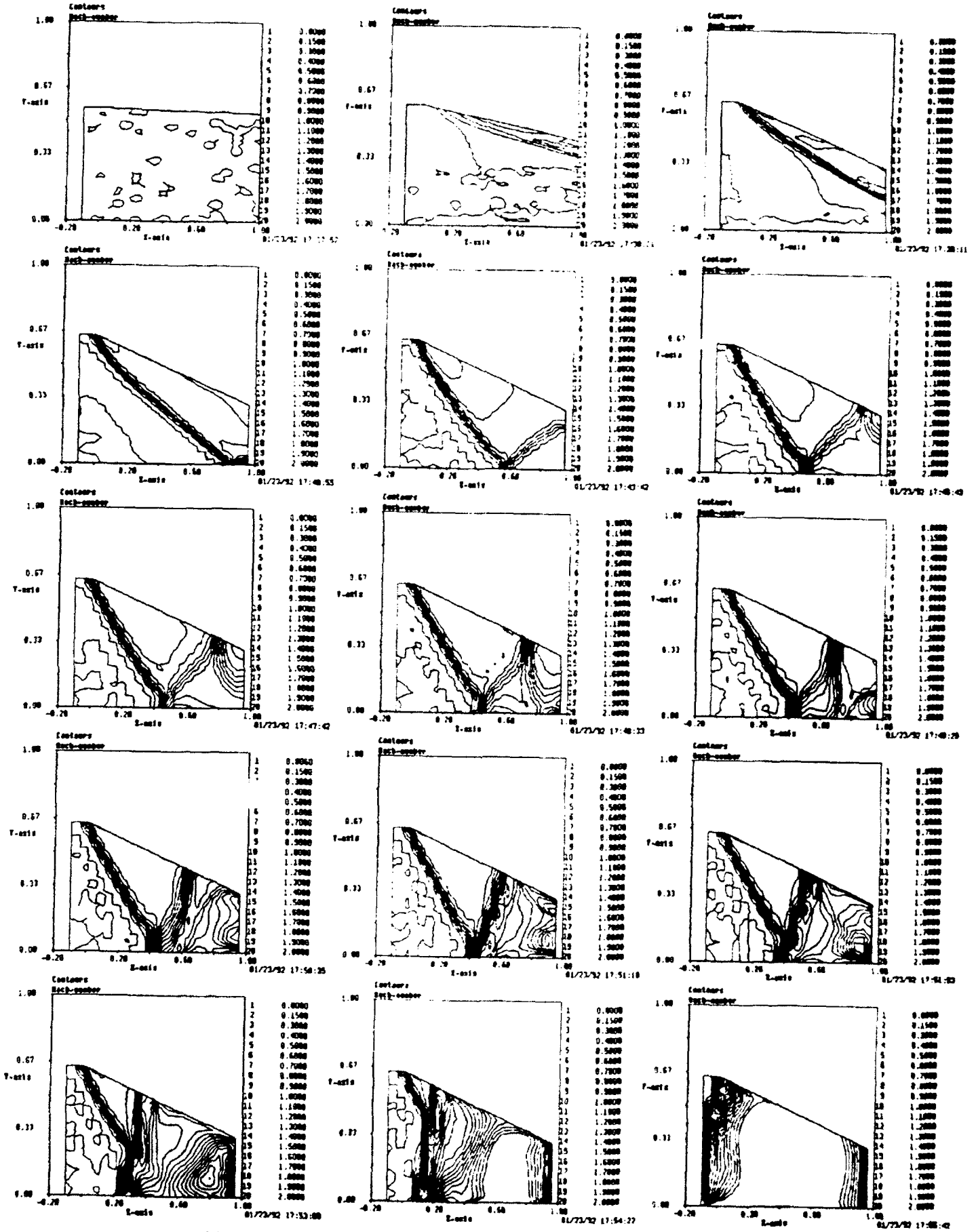


Figure (6). Mach contour history for oscillating wedge flow

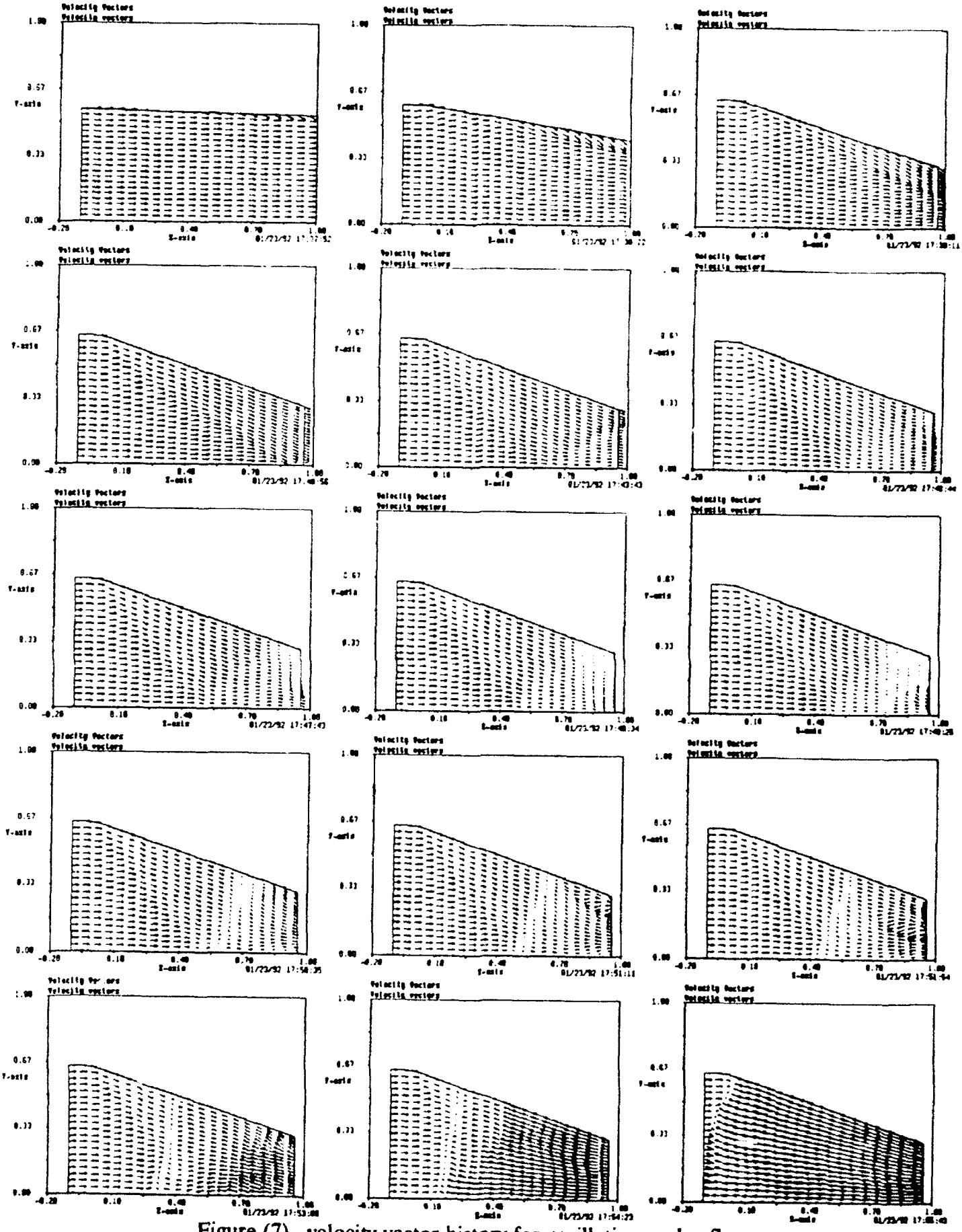


Figure (7). velocity vector history for oscillating wedge flow

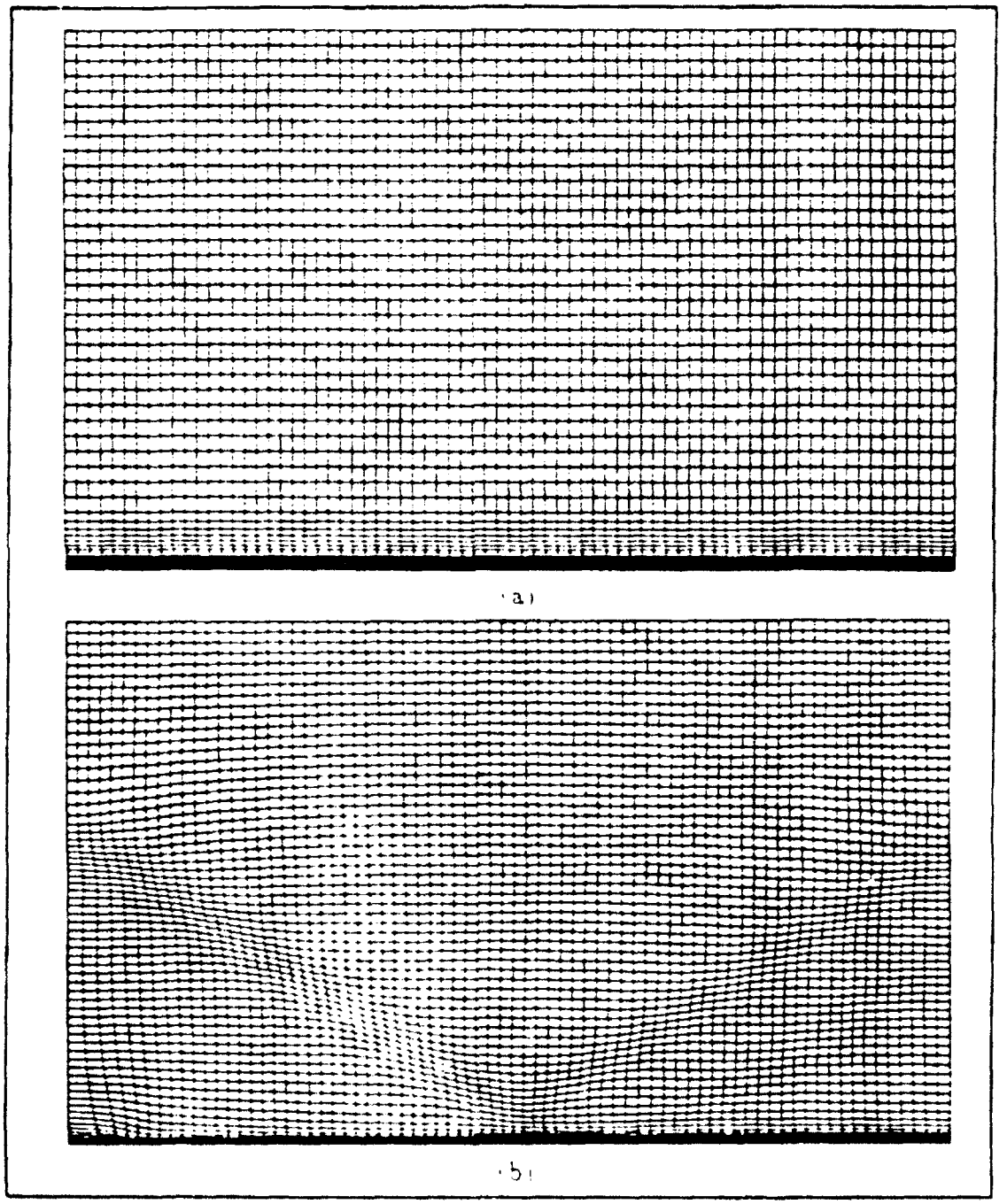


Figure (8). static(a) and dynamically adapted(b) meshes for shock-boundary layer problem

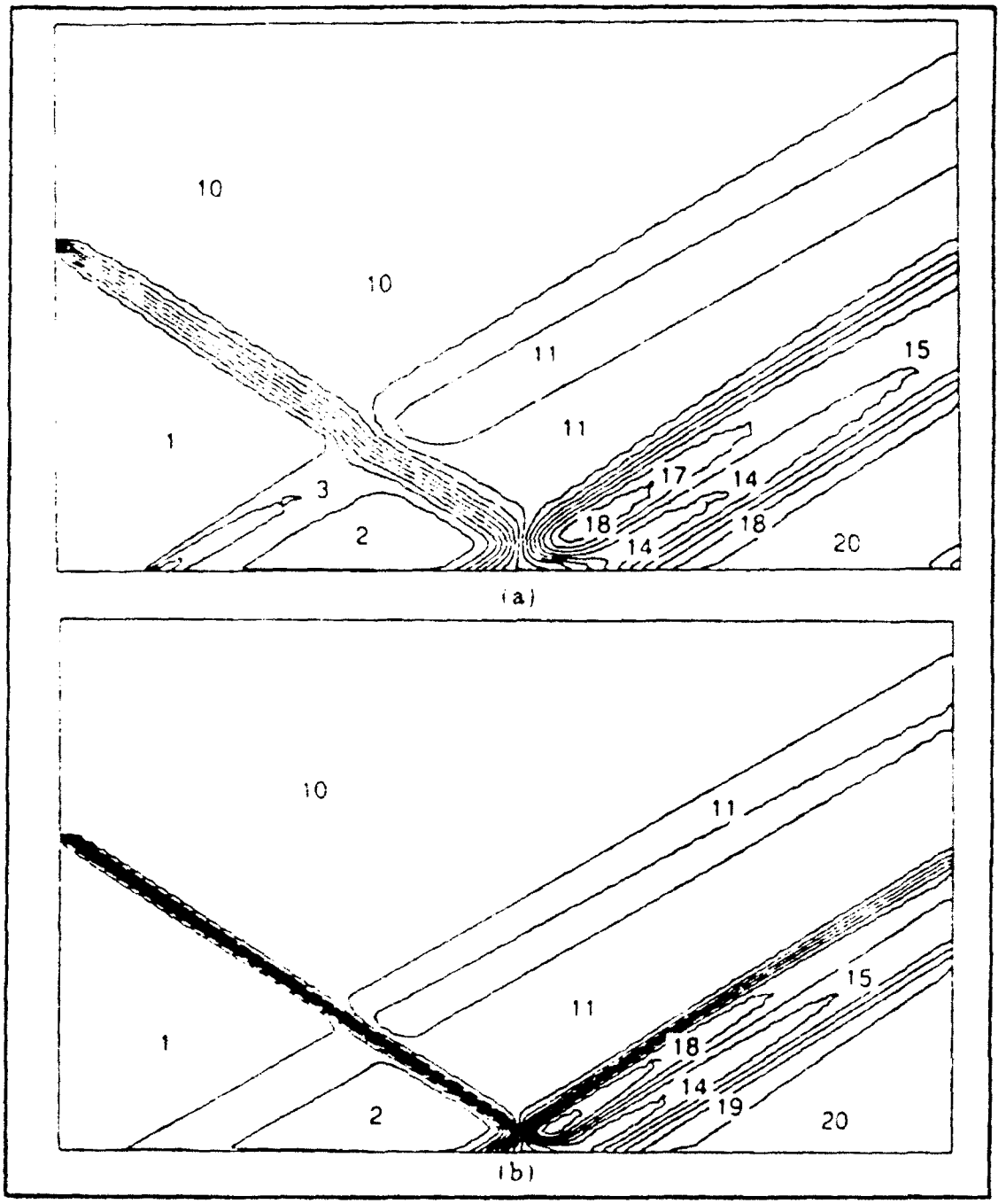


Figure (9). pressure contours for static(a) and dynamically adapted(b) meshes for shock-boundary layer problem

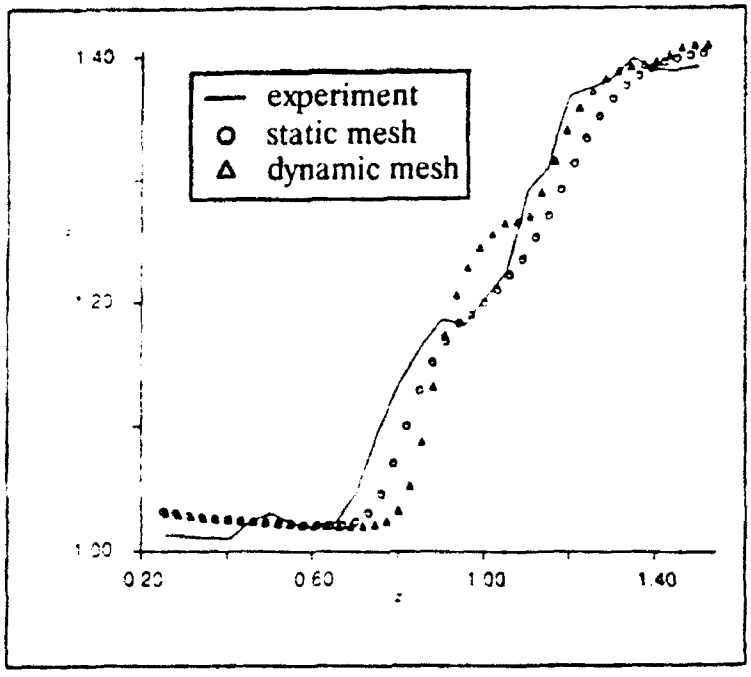


Figure (10). plate pressure coefficient for shock-boundary layer interaction

VIII . APPENDIX

A . Adaptive Mesh On 1-D Scalar Problems

The finite-volume form of the governing equation as applied to a moving control volume has terms involving the grid speed. The solution at the next time step requires the values of the grid speed at each of the mesh points. The grid speed equation, Eq. (2), has the quantity, w_τ, in it, in addition to the primary unknown, x_τ. The w in this equation depends on, say g_1 (see Eq. (3)), which might be something like $(u_\xi)^2$, for example. Thus, the τ derivative of w ultimately depends on u_τ, which in turn depends on, among other things, x_τ. The point is that this coupling between the governing equation and the grid speed equation may be done numerically in more than one way. The next section details the two methods used in this work.

1 . Methods of Coupling

a . Loosely Coupled

In this method, Eq. (2) was discretized by using central differences for all spatial discretization and first order backward differences for w_τ. The discrete system of equations was then solved by a point relaxation scheme. Line relaxation was used on some tests but did not converge as well as the point schemes for several problems. The point schemes were of Jacobi and Gauss–Siedel types, as well as a Newton iteration scheme. The point Newton iteration scheme performed the best overall.

b . Strongly Coupled

This method is quite complex and required extensive coding, even for the 1-D case. It is the most eloquent method of coupling, however. The essence of it is that the dependencies indicated in the open paragraph of VIII A are followed in a most general way. This is done by defining a function, P, as $P = w_\xi/w$. This makes the grid law appear simpler as:

$$x_{\xi\xi} + x_\xi P = 0 . \qquad (10)$$

The corresponding grid speed law is:

$$(x_\tau)_{\xi\xi} + (x_\tau)_\xi P + x_\xi P_\tau = 0 . \qquad (11)$$

P_i represents P at the point i and is expressed as a function of $\vec{x}$ and $\vec{u}$. These vectors represent all of the x and u values in the mesh. Thus, $(P_\tau)_i = (P_i)_{\vec{u}}\vec{u}_\tau + (P_i)_{\vec{x}}\vec{x}_\tau$. The notation on the right hand side of this equation implies a summation of the terms corresponding to each of the elements of the $\vec{x}$ and $\vec{u}$ vectors. Of course, the $\vec{u}_\tau$ is replaced with terms involving $\vec{x}_\tau$ from the governing equation for the problem. The matrices $(P_i)_{\vec{u}}$ and $(P_i)_{\vec{x}}$ are a function of the exact nature of the chosen weight function, w. To complicate matters further, the actual u used in this equation was a smoothed u rather than the computed one. This was necessary

since the computed u often had anomalies due to shocks, etc., which needed to be washed out before using it to drive the mesh. This method produces a grid speed equation which is linear in the unknown $\vec{x}_\tau$, except for the TVD terms. This equation was discretized with central differences for all spatial derivatives except those resulting from the substitution for u_τ. Those were treated identical to their treatment in the discretization of the governing equation. The net result of this procedure was a linear system of equations to solve. This was a nine-diagonal system of equations for which a special solver was written. The bandwidth of this system was a function of how many smoothing passes were used. No smoothing resulted in a pentadiagonal system. The nine-diagonal was a result of two smoothing passes. This system of equations was also solved by various point relaxation schemes with varying degrees of success.

B . Errors Due To Generalized Mappings (1-D)

This section expands section III C and defines the meaning of a "good grid". Suppose a problem of interest requires the solution of a partial differential equation (PDE) which contains a derivative term, f_x, where $f = f(u(x))$ with u(x) representing the dependent variable of the PDE. The numerical solution of such a problem requires first a discretization of the x-domain into a sequence, say, $\{x_i, i = 0,1,....,I\}$ subject to the constraint that $x_0 < x_1 < x_2 < ... < x_I$, and second a suitable numerical approximation to the derivative, f_x. The objective of the choice of the particular sequence $\{x_i\}$ and the particular discretization used in approximating f_x is clearly to minimize the error between the numerical value of f_x and the exact value of f_x at each of the mesh points. Clearly then, a mesh which performs this function is a "good mesh". Moreover, a mesh/discretization combination which produces zero error is the "best mesh" possible.

1 . The Best Mesh (A concept)

Since the same PDE may yield many different solutions depending on the particular initial and boundary conditions imposed, the optimal sequence $\{x_i\}$ (i.e., the best mesh) generally will be different for different problems. In addition, the discrete approximation to f_x could also be different for different problems. This is a common occurrence in many currently popular schemes in which the discretization is determined based on local information in the solution. Upwind schemes and schemes based on a solution's total variation (TV) (e.g., TVD schemes) are examples of this. Prevailing thought, therefore, is usually to first decide upon a discretization procedure for the PDE of interest. Then a procedure is developed for adjusting the mesh point positions to improve the resolution of the computed solution. These are generally treated as separate, if not unrelated, problems. Mesh point adjustment schemes are usually based, in part, on a user's intuition. For example, it is common to find attempts at concentrating mesh points in regions with large values of $|f_x|$ independent of the particular discretization scheme used. Such intuitively driven mesh adjustment schemes work sometimes for some f(x) distributions but not for others. The following analysis helps to explain why.

Given a differentiable function f(x), consider the following finite-difference approximation to the first derivative of this function at the point x_i:

$$f_x(x_i) \approx \frac{f_{i+1} - f_{i-1}}{x_{i+1} - x_{i-1}} \qquad (12)$$

where f_{i+1} means $f(x_{i+1})$, etc. This approximate expression may be written as a strict equality provided the x_i on the left hand side is interpreted as the value of x between x_{i-1} and x_{i+1} at which Eq. (12) becomes exact in the sense of the mean value theorem. Thus

$$f_x(x_i) = \frac{f_{i+1} - f_{i-1}}{x_{i+1} - x_{i-1}} \qquad (13)$$

The problem shaping up here is to find a point, x_i, between each pair of bounding points x_{i-1} and x_{i+1} which causes the finite-difference approximation at the point x_i to produce the exact function derivative at this point. Note that, strictly speaking, the exact function derivative expression is required for such an approach to proceed. Further note that even though the determination of x_i requires an iterative solution to a non-linear equation in general, a unique result is guaranteed by the mean value theorem provided the sequence $\{f_i\}$ on the mesh $\{x_i\}$ adequately represents the true function nature. More specifically, f_{xx} must be of one sign between each pair of bounding points x_{i-1} and x_{i+1}. This requirement presents a problem for intervals containing inflection points except that any given curve can be broken into a sequence of curves, none of which have inflection points on their interior. The break-points of this sequence are, of course, the inflection points themselves.

Since the actual function, $f_x(x)$, is required for the numerical procedure suggested above, it would seem a fruitless procedure for problems of practical interest. However, the value of the procedure lies not specifically in its practical applicability (the extent of which remains to be demonstrated), but rather in its ability to produce categorically the "best mesh" for any differentiable function for which a first derivative expression is known. This will, at a very minimum, provide information about what a "good grid" should look like and what characteristics it should have. This information, however, will be valid only for the particular discrete approximation used in Eq. (13), which is, in this case, a central difference. It is possible to use other approximations also. For example, a second order backward difference would require a solution of the equation:

$$f_x(x_i) = \frac{3f_i - 4f_{i-1} + f_{i-2}}{3x_i - 4x_{i-1} + x_{i-2}} \qquad (14)$$

for the "best" location of the point x_i. The "best mesh" in this case is different than the "best mesh" in the central difference case. The important thing to recognize here is that the "best mesh" depends upon both the *nature of the function* f(x) and the *choice of discretization procedure* of the derivative, f_x. Adaptive mesh procedures to date have focused on only the first of these.

2. Some First Derivative Examples

The actual iterative procedure used for solving Eqs. (13) and (14) is now outlined. Different types of functions are used to illustrate the ability of the method to produce the "best mesh" for both central difference approximations and one-sided difference approximations. As a beginning, consider the iterative form of Eq. (13):

$$\left[x_i^{m+1}\right]^{k+1} = \left[x_i^m\right]^k + \frac{\left(\frac{f(x_{i+1}^k) - f(x_{i-1}^{k+1})}{x_{i+1}^k - x_{i-1}^{k+1}}\right) - \left[f_x\left(x_i^m\right)\right]^k}{\left[f_{xx}\left(x_i^m\right)\right]^k} \qquad (15)$$

This form implies a double iteration nested with the local interval iteration inside of an outer iteration spanning the mesh from lowest index to highest index. This iterative procedure is equivalent to a Newton iteration embedded within a cyclic mesh sweep outer iteration. For each value of the mesh index, i, and starting from an initial condition for the sequence on i, $\{x_i\}^k$ (k = 0), generate x_i^{k+1} as the limit of the convergent sequence on m, $\{x_i^{m+1}\}^{k+1}$. This iterative process, of course, requires f(x) to be of class C^2 but only because of the choice to embed a Newton iteration in the procedure. An equivalent method can be applied to Eq. (14) for a one-sided difference approximation. It should be observed that the procedure must fail if the function f(x) is a straight line, since in that case, it no longer matters where the mesh points are placed from the point of view of evaluating a first derivative with a finite-difference. The resulting difference approximation will always produce the exact derivative regardless of where the mesh points are.

Selected examples of the application of this procedure for various functions are now presented. The examples are selected to demonstrate some significant results.

a. f(x) = sin(πx)

The function sin(πx) serves as the first test case. Figure (1) shows the function along with the two "best mesh" distributions. Note that the points tend to cluster toward the peak region for

both finite-difference approximations. Further note that the two finite-difference approximations produce different "best" meshes and that the mesh for the one-sided scheme is directionally biased as one would expect. A traditional adaptive mesh approach would adapt to some weight function chosen by the user. This might be function gradient, second derivative, curvature, or some combination of these. These three functions are plotted with f(x) in Fig. (2). Note that a weight function based on either f, f_{xx}, or curvature would at least provide the proper clustering direction for this case whereas one based on f_x would not. The determination of the "best mesh", however, requires more than just directional information.

$$b.\ f(x) = ax^4 + bx^3 + cx^2 + dx$$

In this example, $a = .8(p+4)$, $b = -1.6(p+4)$, $c = p$, $d = -.2(p-16)$, where p is a user input parameter which is equal to half of the value of f_{xx} at $x = 0$ and controls the nature of the quartic. The first case is for a user input of $p = -8$. The function along with the two "best mesh" distributions are shown in Fig. (3). This figure clearly illustrates that the points in the "best mesh" for this problem tend to cluster toward the boundaries. Although this function exhibits some of the same qualitative characteristics as the $\sin(\pi x)$ function as seen in Fig. (4), the points in the "best mesh" do the exact opposite of what they do in the quartic case. Obviously, an adaptive mesh procedure which clustered toward, say, high first derivative regions would perhaps improve the resolution of the derivative f_x for the quartic function but would cause poorer resolution for $\sin(\pi x)$.

The final demonstration is made with $p = -4$. Figure (5) shows the "best mesh" for this case while the possible weight functions are shown with f(x) in Fig. (6). Note that for this case, there is no clustering at all for either of the finite-difference approximations. It turns out that this case actually corresponds to a quadratic function rather than a quartic. This fact coupled with the use of 2nd order finite-difference approximations provides a situation in which exact derivatives are produced on an equally spaced mesh. Moreover, it is observed that when f_{xx} of the function, f(x), exceeds that of the parabolic function, points tend to cluster toward the center while if f_{xx} is less than that of the parabolic function, points cluster toward the boundaries. This information may be used in future mesh adaption schemes to improve the ability of the scheme to generate a "better mesh" with assurance of improving accuracy.

3. Other Possibilities

In theory, one could determine the "best mesh" to resolve numerical approximations to other types of terms also, such as the quantity: $f_x - \mu u_{xx}$ found in the viscous Burgers' equation. In this case, however, other conditions besides no inflection points must likely be imposed to guarantee uniqueness of the result, if such uniqueness is even possible.

The practical usefulness of the procedure outlined here is still unknown for multi-dimensional multi-variable problems. However, the lessons learned are clearly valuable guides in helping to formulate new weight functions and new adaption schemes which incorporate informa-

tion regarding both the *nature of the function* f(x) and the *discrete approximation used* to represent its derivatives.

C . 2-D Considerations

New considerations are important in extending to multi-dimensional problems. One of the most obvious is the issue of orthogonality. The basic concern is the error introduced into a computation as a result of non-orthogonality of the mesh. In a dynamically adapting mesh, this concern is clearly greater since the real possibility exists that cell collapse may be inadvertently driven by the adaption scheme. An attempt was made to study the way in which non-orthogonality influences the numerical approximation. The basic procedure is outlined with the key results highlighted.

1 . Errors Associated With Mesh Non-Orthogonality

a . Procedure

Consider an equation of the form $u_t + f_x + g_y = 0$. An application of a generalized mapping from (t,x,y) to (τ,ξ,η) results in a typical term, say f_x, being replaced by its analytic equivalent given by: $f_x = \xi_x f_\xi + \eta_x f_\eta$ or $f_x = \frac{y_\eta f_\xi - y_\xi f_\eta}{J}$ where $J = x_\xi y_\eta - y_\eta x_\xi$. If the quantities, f_ξ and f_η, are approximated by central differences, the following expression results:

$$f_x \approx \frac{y_\eta}{2J\Delta\xi}\left(f_{i+1} - f_{i-1}\right) - \frac{y_\xi}{2J\Delta\eta}\left(f_{j+1} - f_{j-1}\right)$$

The next step is to apply a 2-D Taylor expansion to each f term and then collect all like derivative terms together resulting in:

$$f_x \approx \left\{\frac{\Delta\eta\ y_\eta\ (\hat{x}_{i+1} - \hat{x}_{i-1}) - \Delta\xi\ y_\xi\left(\hat{x}_{j+1} - \hat{x}_{j-1}\right)}{2J\Delta\xi\Delta\eta}\right\} f_x$$

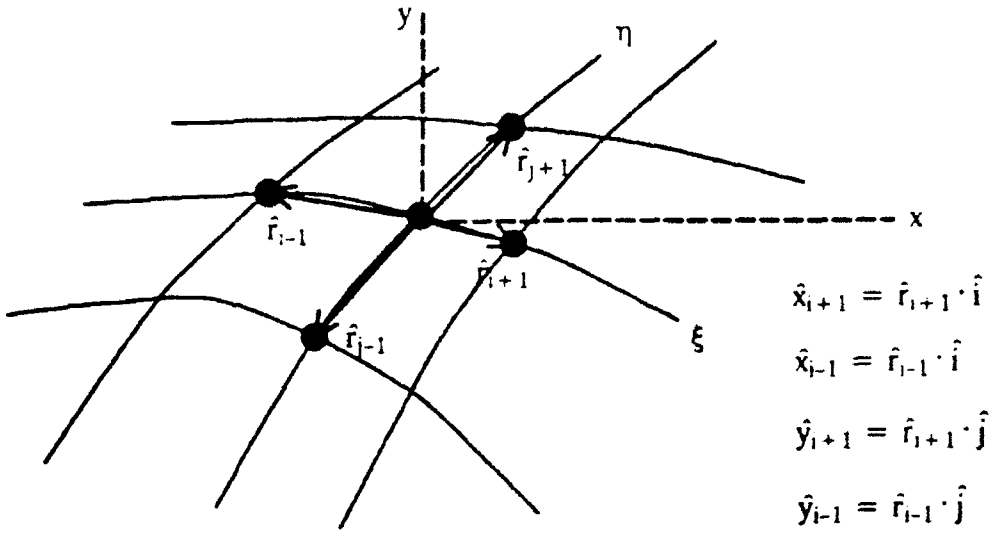
$$+ \left\{\frac{\Delta\eta\ y_\eta\ (\hat{y}_{i+1} - \hat{y}_{i-1}) - \Delta\xi\ y_\xi\left(\hat{y}_{j+1} - \hat{y}_{j-1}\right)}{2J\Delta\xi\Delta\eta}\right\} f_y$$

$$+ \left\{\frac{\Delta\eta\ y_\eta\ \left(\hat{x}^2_{i+1} - \hat{x}^2_{i-1}\right) - \Delta\xi\ y_\xi\left(\hat{x}^2_{j+1} - \hat{x}^2_{j-1}\right)}{4J\Delta\xi\Delta\eta}\right\} f_{xx}$$

$$+ \left\{\frac{\Delta\eta\ y_\eta\ \left(\hat{y}^2_{i+1} - \hat{y}^2_{i-1}\right) - \Delta\xi\ y_\xi\left(\hat{y}^2_{j+1} - \hat{y}^2_{j-1}\right)}{4J\Delta\xi\Delta\eta}\right\} f_{yy}$$

$$+\left\{\frac{\Delta\eta\ y_\eta\ (\hat{x}_{i+1}\hat{y}_{i+1}-\hat{x}_{i-1}\hat{y}_{i-1})-\Delta\xi\ y_\xi\left(\hat{x}_{j+1}\hat{y}_{j+1}-\hat{x}_{j-1}\hat{y}_{j-1}\right)}{2J\Delta\xi\Delta\eta}\right\}f_{xy}+\ldots$$

where $\hat{x}$, $\hat{y}$ are defined in the sketch below.



b. Analysis

Let this be expressed as $C_1f_x + C_2f_y + C_3f_{xx} + C_4f_{yy} + C_5f_{xy} + \ldots$ Then it is desirable to have $C_1 = 1$ while the other C's are as small as possible. It is easy to show that $C_1 = 1$ and $C_2 = 0$ if $x_\xi, x_\eta, y_\xi, y_\eta$ are determined from the central difference approximations:

$$x_\xi \approx \frac{\delta_i(x)}{2\Delta\xi} = \frac{\hat{x}_{i+1}-\hat{x}_{i-1}}{2\Delta\xi}, \qquad x_\eta \approx \frac{\delta_j(x)}{2\Delta\eta} = \frac{\hat{x}_{j+1}-\hat{x}_{j-1}}{2\Delta\eta},$$

$$y_\xi \approx \frac{\delta_i(y)}{2\Delta\xi} = \frac{\hat{y}_{i+1}-\hat{y}_{i-1}}{2\Delta\xi}, \qquad y_\eta \approx \frac{\delta_j(y)}{2\Delta\eta} = \frac{\hat{y}_{j+1}-\hat{y}_{j-1}}{2\Delta\eta}.$$

With these metric evaluations and the second difference definitions,

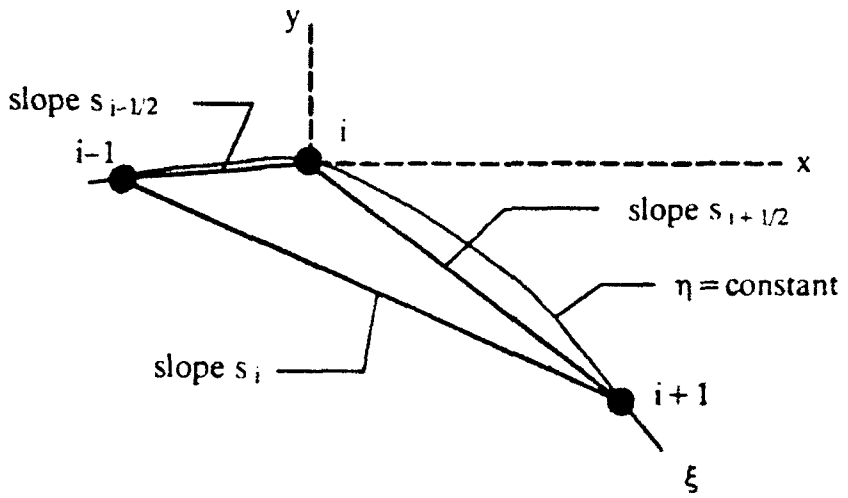
$\delta_i^2(x) = \hat{x}_{i+1} + \hat{x}_{i-1}$, $\delta_j^2(x) = \hat{x}_{j+1} + \hat{x}_{j-1}$, $\delta_i^2(y) = \hat{y}_{i+1} + \hat{y}_{i-1}$, and $\delta_j^2(y) = \hat{y}_{j+1} + \hat{y}_{j-1}$,

the C_3 and C_4 are expressed as:

$$C_3 = \frac{\delta_j(y)\delta_i(x)\delta_i^2(x) - \delta_i(y)\delta_j(x)\delta_j^2(x)}{2\left[\delta_i(x)\delta_j(y) - \delta_j(x)\delta_i(y)\right]}, \qquad C_4 = \frac{\delta_j(y)\delta_i(y)\delta_i^2(y) - \delta_i(y)\delta_j(y)\delta_j^2(y)}{2\left[\delta_i(x)\delta_j(y) - \delta_j(x)\delta_i(y)\right]}$$

The values of these C's are small if the numerators are small and/or the denominators are large. Consider first the C_3 coefficient. The numerator of this coefficient is zero when the equality, $p_\xi = p_\eta$ is realized. In this equality, each of the p's represents a percent difference in the slope of the corresponding coordinate line between the + and - cells. For example, p_ξ

represents the difference between the slope of the η = constant line connecting i with i + 1 and i with i–1 divided by the slope of the line connecting i + 1 with i–1. This is expressed as $p_\xi = \frac{s_{i+\frac{1}{2}} - s_{i-\frac{1}{2}}}{s_i}$. See the sketch below.



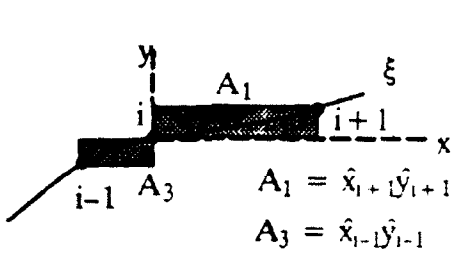
Since this is not likely to happen for any mesh, let alone a dynamically adaptive one, it is safe to assume that C_3 will be non–zero. Thus it is even more important that the denominator is not too small. Close examination of the denominator reveals that it is proportional to the cross product of the two vectors connecting i + 1 with i–1 and j + 1 with j–1. This means that the denominator is proportional to the product of the magnitudes of these two vectors and the sine of the angle between them. This is where non–orthogonality can be detrimental to the accuracy of the approximation. As the mesh tends to extreme non–orthogonality, the denominator tends to zero.

Continuing with the C_4 coefficient, it is clear that if the numerator is not zero, the significance of the denominator is identical to that indicated for C_3. The numerator is seen to be zero for the most general case when $\delta_i^2(y) = \delta_j^2(y)$. Again, this is not at all likely and consequently the effect of non–orthogonality can be significant.

The C_5 coefficient has a little different interpretation. Since it multiplies a cross derivative, one would expect the coefficient to have something to do with areas. C_5 can be written as:

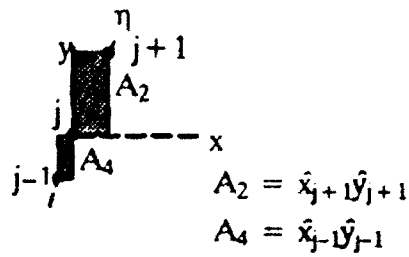
$$C_5 = \frac{\delta_j(y)\left[\hat{x}_{i+1}\hat{y}_{i+1} - \hat{x}_{i-1}\hat{y}_{i-1}\right] - \delta_i(y)\left[\hat{x}_{j+1}\hat{y}_{j+1} - \hat{x}_{j-1}\hat{y}_{j-1}\right]}{\left[\delta_i(x)\delta_j(y) - \delta_j(x)\delta_i(y)\right]}$$

From the following two sketches, the areas $A_1 - A_4$ are defined. C_5 is then expressed in terms of these areas.



$$A_1 = \hat{x}_{i+1}\hat{y}_{i+1}$$

$$A_3 = \hat{x}_{i-1}\hat{y}_{i-1}$$



$$A_2 = \hat{x}_{j+1}\hat{y}_{j+1}$$

$$A_4 = \hat{x}_{j-1}\hat{y}_{j-1}$$

$$C_5 = \frac{\delta_j(y)[A_1 - A_3] - \delta_i(y)[A_2 - A_4]}{\left[\delta_i(x)\delta_j(y) - \delta_j(x)\delta_i(y)\right]}$$

From this interpretation, it is clear that the numerator of the C_5 coefficient is not likely to vanish, except in special situations, and consequently the significance of the size of the denominator is amplified. Again, this depends on the extent of the non-orthogonality of the mesh.

In summary, the connection between non-orthogonality and error has been established for the case of central differences. The basic results are expected to be applicable to upwind difference approximations as well since they appear to be common sense in nature. The basic requirements are:

1. metrics should be evaluated the same as f_ξ , f_η .
2. minimizing coordinate line curvature and stretching also minimizes numerators of C_3, C_4, C_5.
3. minimizing non-orthogonality maximizes denominators of C_3, C_4, C_5.

D . Dynamic Adaption For 2-D Euler Equations

The basic grid and grid-speed laws for 2-D problems introduced by [1] are dimensionally inconsistent unless dimensions are absorbed into the user input parameters. The cure for this scaling problem was to redefine the performance index in a dimensionally consistent manner with user input parameters of order 1. The new performance index is given as:

$$I = \int\int L(\vec{r},\ \vec{r}_\xi,\ \vec{r}_\eta)d\xi d\eta \qquad (16)$$

where $L = L_s + L_o + L_a$ and:

$$L_s = \lambda_s \frac{\vec{r}_\xi \cdot \vec{r}_\xi + \vec{r}_\eta \cdot \vec{r}_\eta}{J}$$

$$L_o = \lambda_o \left(\frac{\vec{r}_\xi \cdot \vec{r}_\eta}{J} \right)^2$$

$$L_a = \lambda_a \left(\frac{wJ}{K} \right)^2$$

$$K = \frac{\int\int w dx dy}{(\xi_{max} - \xi_{min})(\eta_{max} - \eta_{min})}$$

The Euler-Lagrange equations were then re-derived for this case and the grid speed equations were derived by differentiating these new Euler-Lagrange equations with respect to the temporal variable.

1 . Modified Mesh Law

The result of the derivation was a second order PDE system of the form:

$$G(\vec{r}) = A\vec{r}_{\xi\xi} + B\vec{r}_{\xi\eta} + C\vec{r}_{\eta\eta} + D\vec{r}_\xi + E\vec{r}_\eta = 0 \qquad (17)$$

which constitutes the grid law. The boundary conditions are mixed Dirichlet and Neumann depending upon the physical conditions required of the mesh at the boundaries.

In this equation, $A = A_s + A_o + A_a$, $B = B_s + B_o + B_a$, etc. These matrices are given as follows:

$$A_s = \frac{\lambda_s a}{J^3} S \quad , \quad B_s = -\frac{2\lambda_s \beta}{J^3} S \quad , \quad C_s = \frac{\lambda_s \gamma}{J^3} S \quad , \quad D_s = 0 \quad , \quad E_s = 0$$

$$S = \begin{bmatrix} y_\xi^2 + y_\eta^2 & -(x_\xi y_\xi + x_\eta y_\eta) \\ -(x_\xi y_\xi + x_\eta y_\eta) & x_\xi^2 + x_\eta^2 \end{bmatrix} = \begin{bmatrix} \delta & -\epsilon \\ -\epsilon & \varrho \end{bmatrix} \qquad J = x_\xi y_\eta - x_\eta y_\xi$$

$$a = x_\eta^2 + y_\eta^2 \qquad \beta = x_\xi x_\eta + y_\eta y_\xi \qquad \gamma = x_\xi^2 + y_\xi^2$$

$$A_o = \frac{\lambda_o a}{J^4} \begin{bmatrix} y_\xi(2\beta y_\eta + a y_\xi) & -(\gamma x_\eta y_\eta + 2a x_\xi y_\xi) \\ -(\gamma x_\eta y_\eta + 2a x_\xi y_\xi) & x_\xi(2\beta x_\eta + a x_\xi) \end{bmatrix}$$

$$B_o = -\frac{\lambda_o(2\beta^2 + a\gamma)}{J^4} \begin{bmatrix} 2y_\xi y_\eta & -\phi \\ -\phi & 2x_\xi x_\eta \end{bmatrix} \qquad \phi = x_\xi y_\eta + x_\eta y_\xi$$

$$C_o = \frac{\lambda_o \gamma}{J^4} \begin{bmatrix} y_\eta(2\beta y_\xi + \gamma y_\eta) & -(a x_\xi y_\xi + 2\gamma x_\eta y_\eta) \\ -(a x_\xi y_\xi + 2\gamma x_\eta y_\eta) & x_\eta(2\beta x_\xi + \gamma x_\eta) \end{bmatrix}$$

$$D_o = 0 \quad , \quad E_o = 0$$

$$A_a = \frac{\lambda_a w^2}{K^2} \begin{bmatrix} y_\eta^2 & -x_\eta y_\eta \\ -x_\eta y_\eta & x_\eta^2 \end{bmatrix} \quad , \quad B_a = \frac{\lambda_a w^2}{K^2} \begin{bmatrix} -2y_\xi y_\eta & \phi \\ \phi & -2x_\xi x_\eta \end{bmatrix}$$

$$C_a = \frac{\lambda_a w^2}{K^2} \begin{bmatrix} y_\xi^2 & -x_\xi y_\xi \\ -x_\xi y_\xi & x_\xi^2 \end{bmatrix} \quad , \quad D_a = \frac{\lambda_a J w w_\eta}{K^2} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

$$E_a = \frac{\lambda_a J w w_\xi}{K^2} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

This new mesh law requires specification of 3 parameters, λ_s, λ_o, and λ_a corresponding to smoothness, orthogonality, and adaption terms. Note that in the new formulation, these may be functions of time. This may be useful for problems in which the flow structure being adapted to diminishes with time while the user wishes to maintain resolution at this location during the decay process. These parameters are nondimensional and have the value of unity if orthogonality and solution adaption are equal in importance to smoothness. In the old scheme, the values of these parameters were functions of the problem, the number of mesh

points, the distribution of points, etc., which required experimentation with various values in order to discover an acceptable set (if one even existed).

2. Corresponding Mesh Speed Law

The result of the temporal differentiation was a linear, variable coefficient, second order PDE system of the form:

$$G_\tau(\vec{r}) = A\vec{z}_{\xi\xi} + B\vec{z}_{\xi\eta} + C\vec{z}_{\eta\eta} + D^*\vec{z}_\xi + E^*\vec{z}_\eta + \vec{T}^* = 0 \qquad (18)$$

which constitutes the grid speed law. In this equation, $\vec{z} = \vec{r}_\tau$, the grid speed vector. The boundary conditions are mixed Dirichlet and Neumann for this equation as well, depending upon the physical conditions required of the mesh speed at the boundaries.

The matrices D^* and E^* and the vector $\vec{T}^*$ are quite complicated and involve several pages of definition. They result from differentiating A, B, C, D, E, J, α, β, γ, ϕ, and w with respect to τ, and then expressing the result as a matrix times the grid speed vector derivatives where possible.

E . Appendix Figures

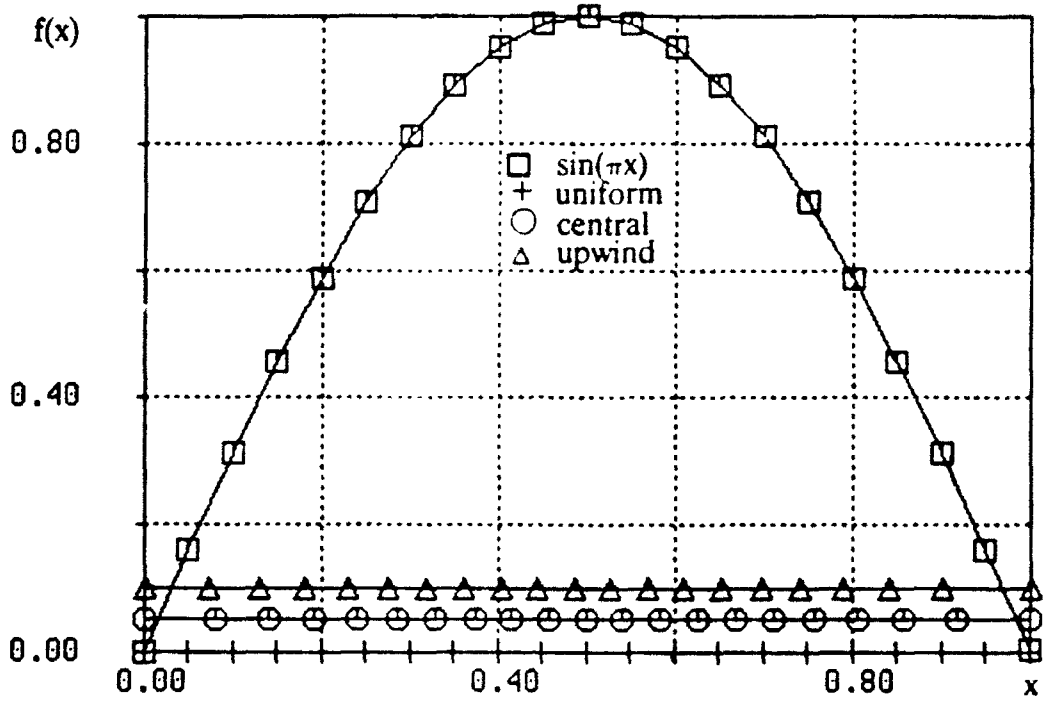


Figure (1). f(x) and mesh position for uniform, "best" central and "best" one-sided.

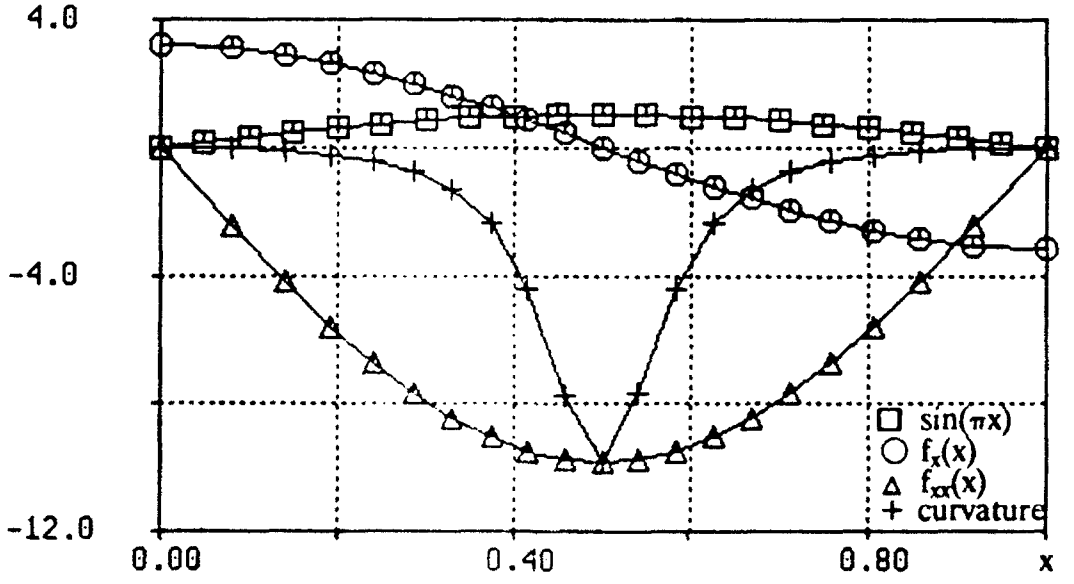


Figure (2). f(x), $f_x(x)$, $f_{xx}(x)$, and curvature for sin(πx).

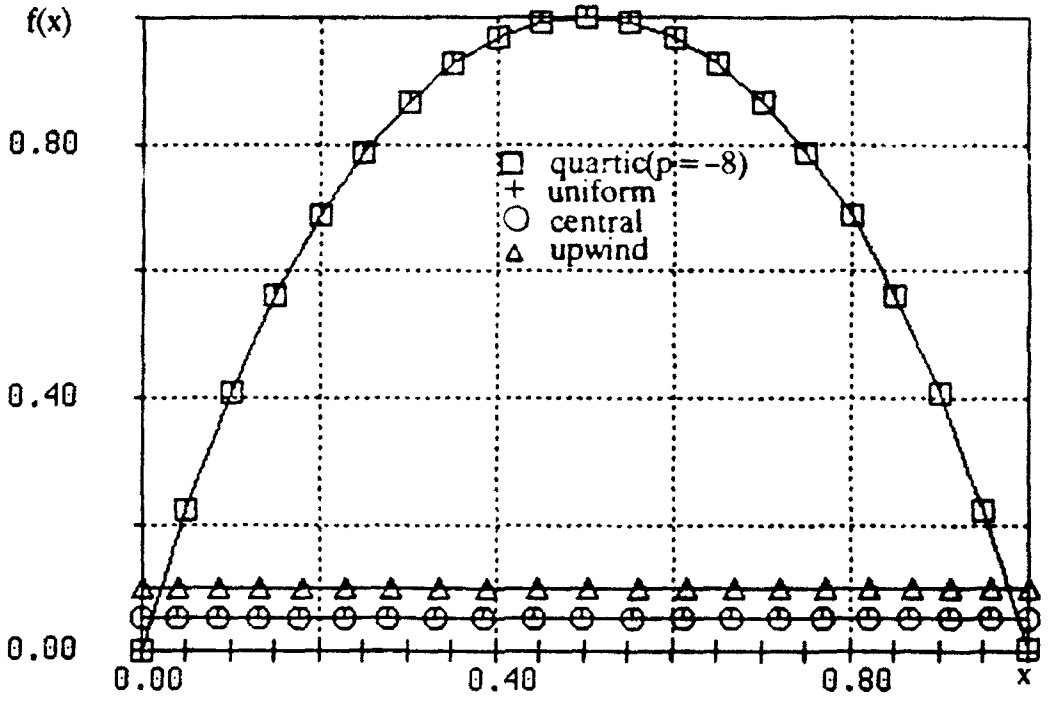


Figure (3). f(x) and mesh position for uniform, "best" central and "best" one-sided.

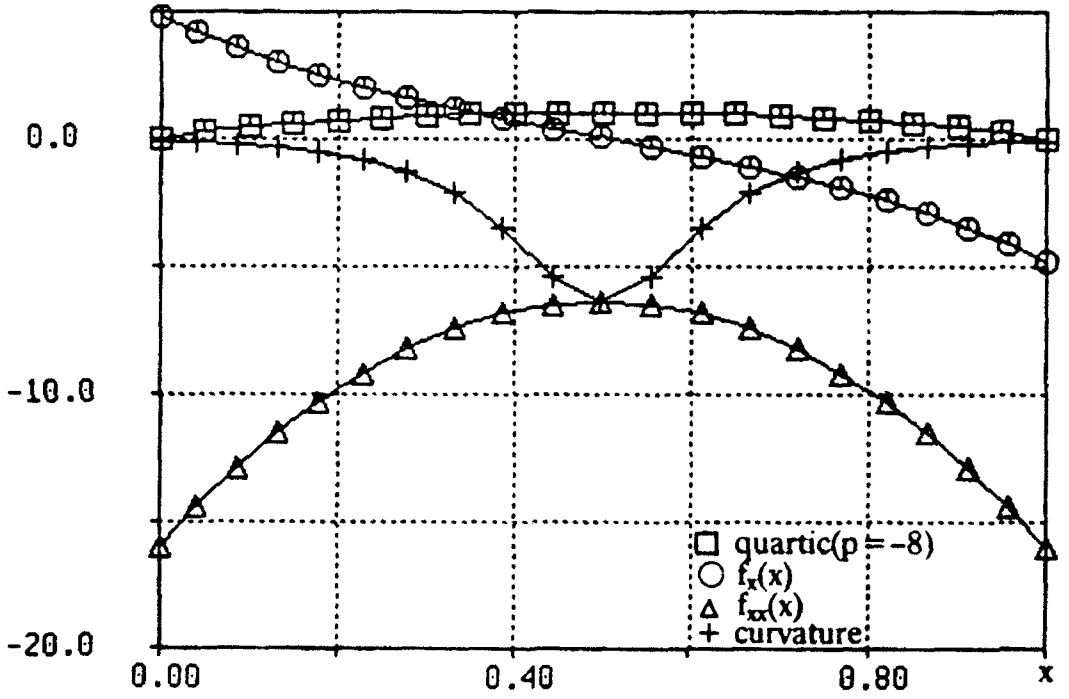


Figure (4). f(x), $f_x(x)$, $f_{xx}(x)$, and curvature for quartic(p=-8).

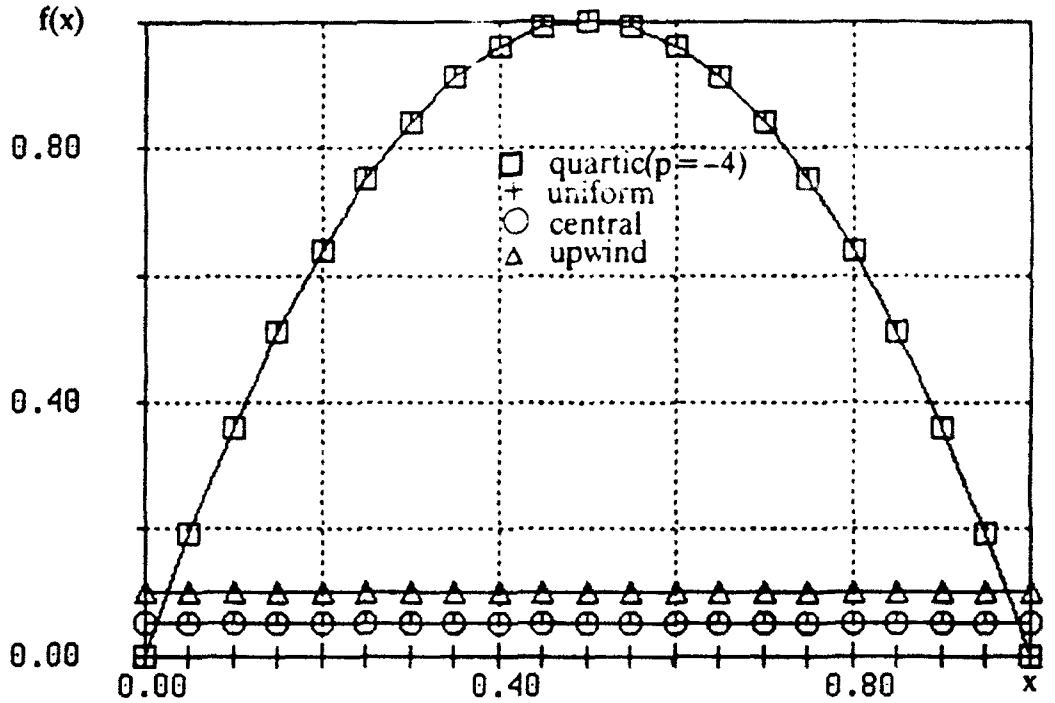


Figure (5). f(x) and mesh position for uniform, "best" central and "best" one-sided.

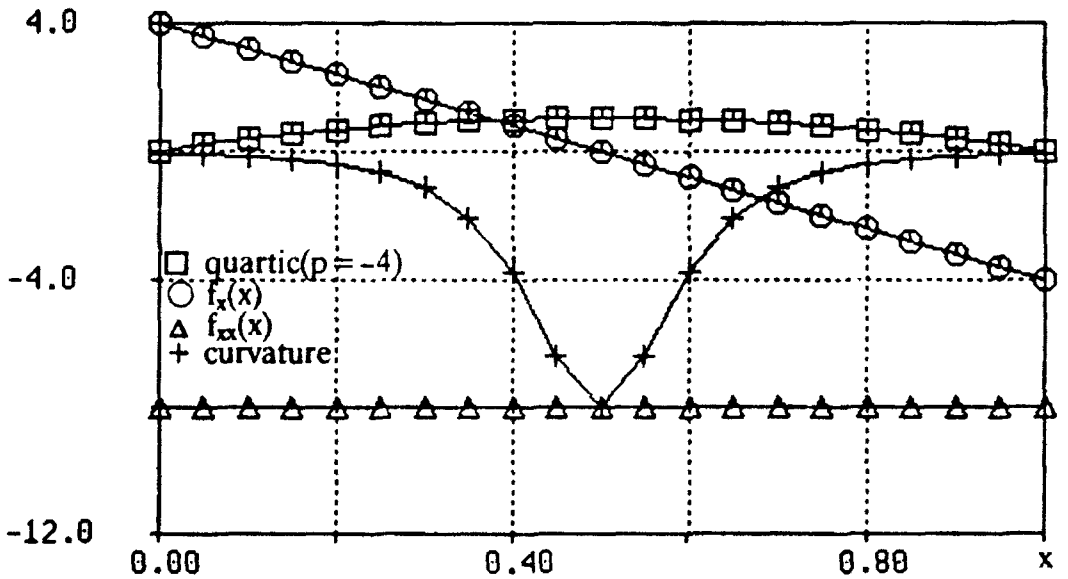


Figure (6). f(x), $f_x(x)$, $f_{xx}(x)$, and curvature for quartic(p = -4).